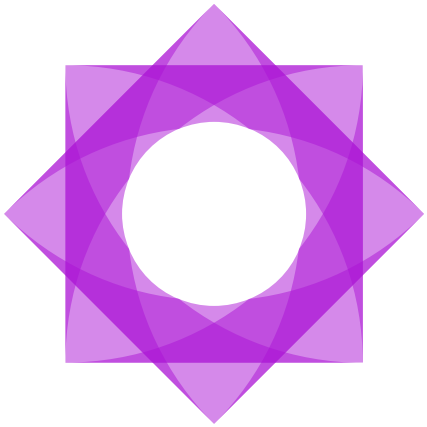


Lasernet 10.

Lasernet Developer 10

Adam McStravick, Torben Pedersen, Sunil Panchal
Revision 17
2026-01-27

Formpipe.

Formpipe.

Contents.

1 Introduction.

1.1 Who Should Use This Guide?

This guide is written for Lasernet Developers. It is intended primarily as a reference to the various functions of Lasernet. It explains the purpose and operation of each module and modifier and how to use them effectively.

2 Terms of Use.

No part of this publication may be reproduced, transmitted, transcribed, or translated into any language in any form by any means without the prior written permission of Formpipe Software. The information in this manual is subject to change without notice. Any company names or data is fictive unless otherwise stated.

Formpipe Software shall not be liable for any loss or damage whatsoever arising from the use of this manual and the information contained therein (including errors or omissions).

Trademarks of other companies mentioned in this document appear for identification purposes only and are the property of their respective companies.

3 Lasernet Developer.

3.1 Creating Your First Lasernet Configuration

The following sections describe how to create your own Lasernet configuration from the start. If you upgrade from Lasernet 9 to Lasernet 10 there is no need to manually import/export anything to get up and running.

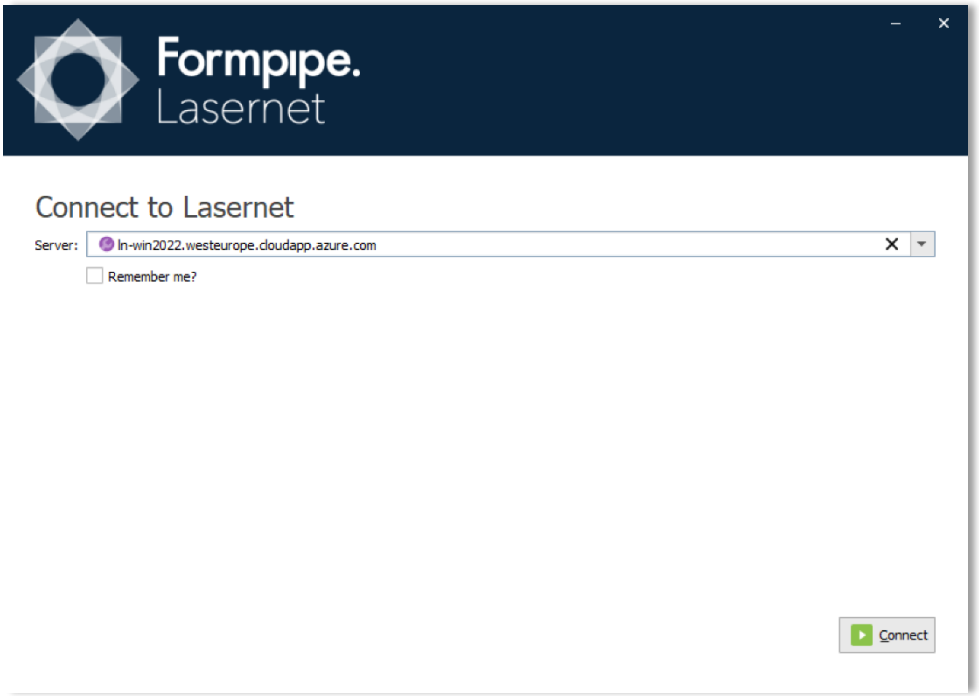
3.2 Getting Started

Lasernet Developer provides an easy-to-use environment for configuring and maintaining input and output management rules on your Lasernet Server and designing great looking forms.

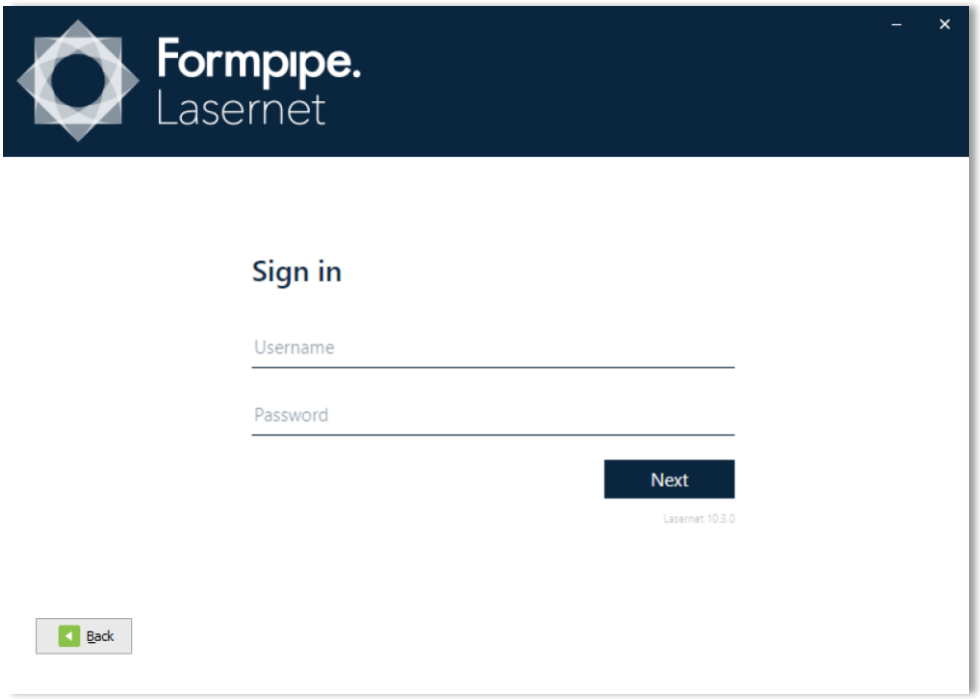
This chapter will give you an introduction to the various functions of the user interface.

3.2.1 Connect to Lasernet

The **Connect to Lasernet** login screen provides secure access to the Lasernet Config Server. The configuration server stores all your configurations, server settings, history, users/groups and security roles.

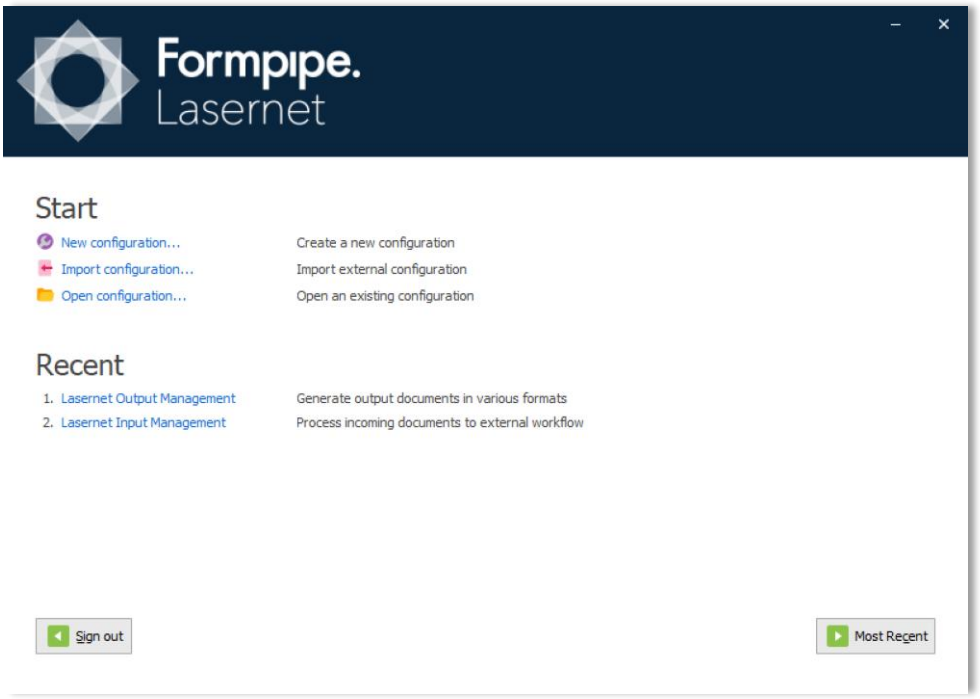


Server	Name of the Lasernet Configuration Server hosted on-premise or in cloud.
Remember me	Activate to remember the last used credentials.



Username	A user name added to the Lasernet Configuration Server.
Password	The password for the given username

3.2.2 Start

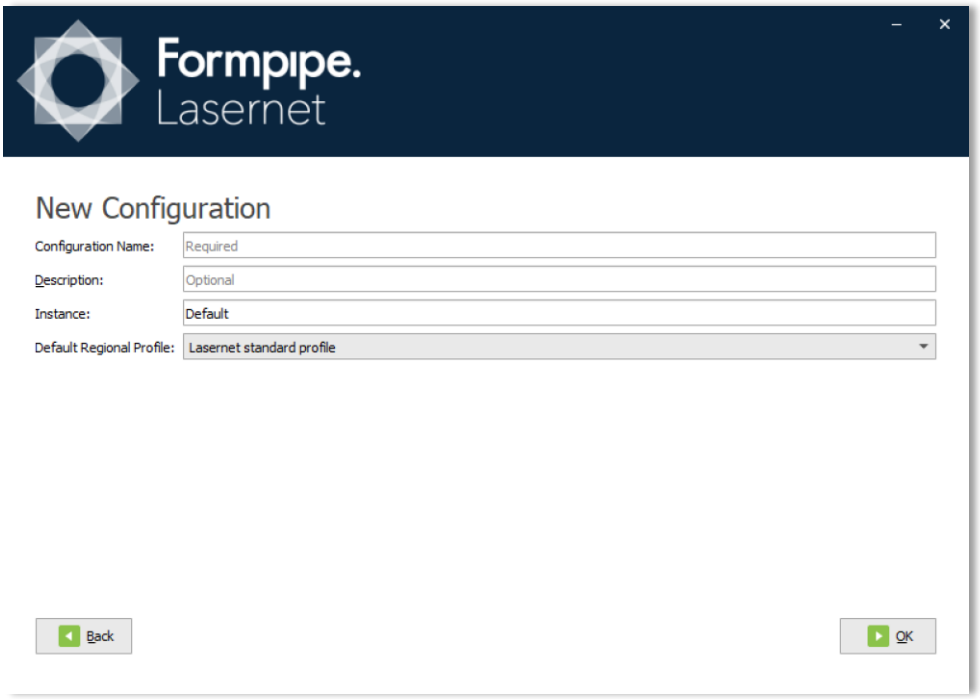


New configuration	Create a new Lasernet configuration.
Import configuration	Import a configuration from disk.
Open configuration	Open a configuration stored in the Lasernet Config 10 server.
Recent	Open a recently used configuration.

Click **Back** to return to the **Connect to Lasernet** screen. Click **Most Recent** to open the most recently used configuration.

3.2.3 New Configuration

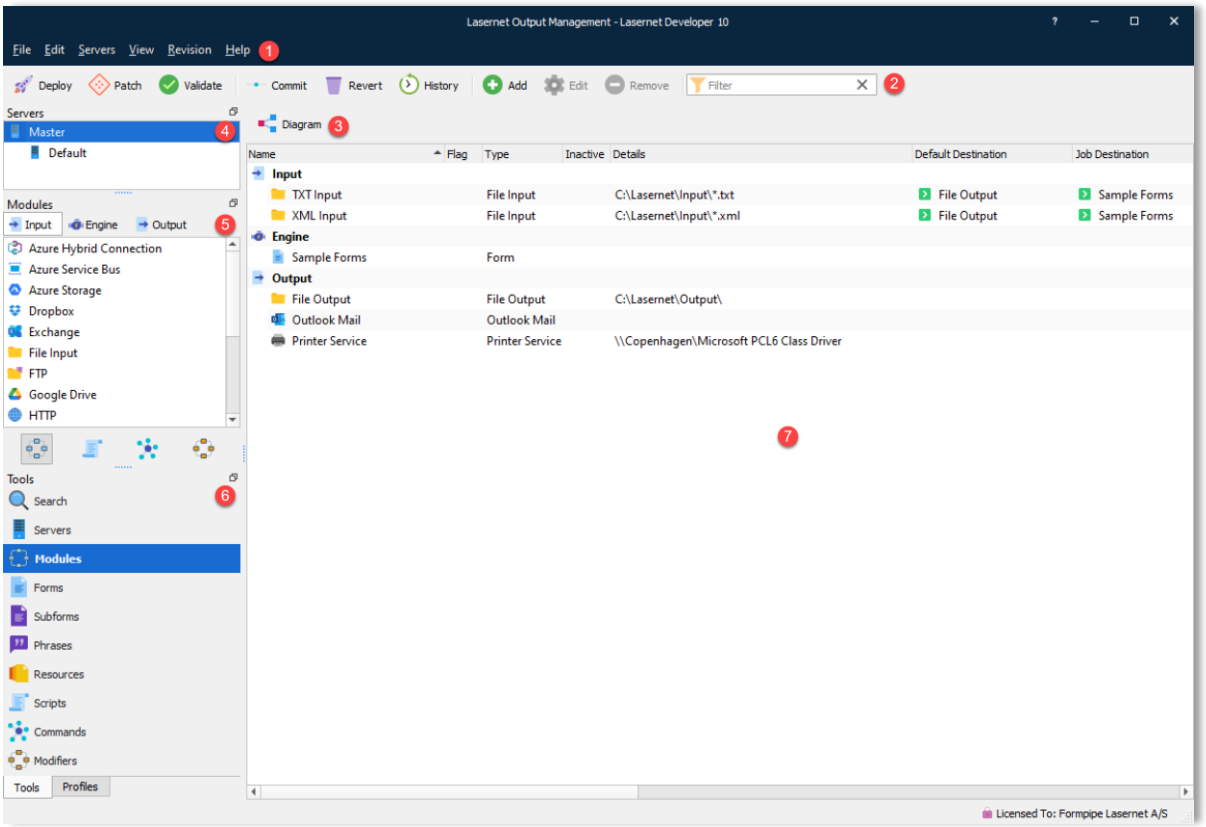
Create a new Lasernet configuration.



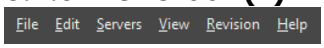
Configuration Name	Name of the configuration to be stored on the configuration server.
Description	Type a description for the configuration.
Instance	Instance name of server that runs the Lasernet service as configured on the Lasernet Config Server.
Default Regional Profile	Select the default region for your configuration. This is used for formatting numbers and dates in documents. Select “Lasernet standard profile” to obtain settings from the local computer. Additional regional profiles can be added to the configuration at a later stage.

3.2.4 User Interface Overview

The Lasernet Developer brings together a number of comprehensive tools that enable a developer to: set up documents, work with data, set business rules and logic, communicate between modules and environments (mail server, web server, printers, etc) and use design features. Below is a brief list of the different tools.



3.2.5 Menu bar (1)



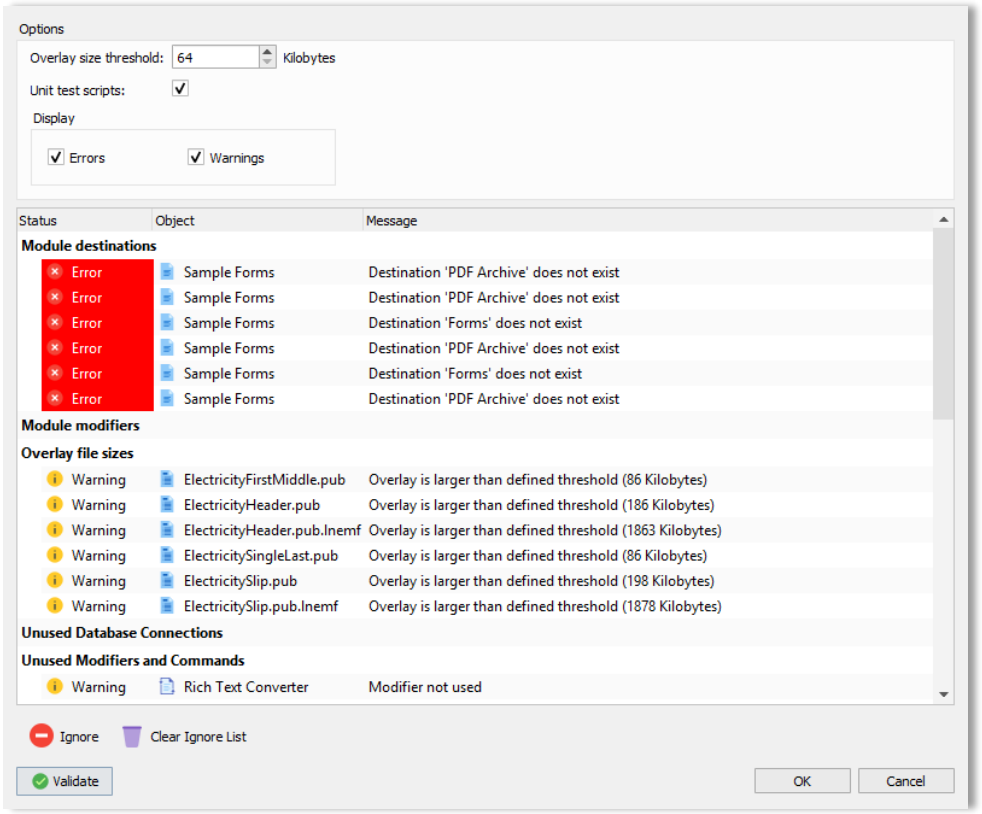
File → Start Page
Return to the Start Page and connect to a new server, login as a new user or open a new configuration.

File → Export Configuration
Export the full configuration as an .lnconfigx file. Import the .lnconfigx file from the Start Page → Import Configuration dialog.

File → Export/Import Objects
More information regarding Export/Import Objects is available in *Chapter 1.4 Export/Import of objects*.

File → Deploy/Patch
More information regarding Deploy/Patch Objects is available in the manual *Lasernet Configuration Server and Deployment (Lasernet Config & Deployment)*.

File → Validate
Select **Validate** to parse the configuration. Lasernet Developer will check your configuration and inform you about a list of known warnings/errors found in the configuration.



Module destinations Checks for orphaned module objects. If a destination has been added to a module that does not exist an error for the given module will be listed.

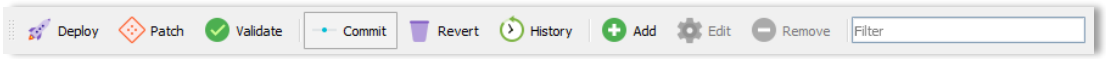
Module modifiers Checks if unknown modifiers are called. If an unknown modifier has been added to a module, an error will list the name of the unknown modifier, the event point name and module name.

Overlay file size Checks for overlay file sizes greater than the user defined threshold. If large overlay files are added to forms, it can result in big EMF and PDF files which can slow down processing. The default threshold value is set to 64 kilobytes. The recommended value can vary between configurations depending on output. If processing a batch of documents containing lots of pages, we recommend using a low threshold value (64kb). If you are only printing single jobs, it is possible to set a high value (1024 kb).

Unused Database Connections Checks for unused database connections in the configuration. This alert is provided for your information only, as existing database connections that are not being used can often be deleted.

Unused Modifiers and Database Commands	Checks for unused modifiers and database commands in the configuration. This alert is provided for your information only as unused existing modifiers and database commands can often be deleted. **Important:** The validation check will not include modifiers and database commands executed as script commands.
Rearrange object names and script names	Checks for conflicting names between rearrange objects and script functions. Both will be seen as objects by the JavaScript engine so the same name cannot be used twice.
Script	Checks for unknown and inactive script modifiers. Will list the name of the unknown script modifier, the event point name and script name.
Script Unit Test	Runs the defined unit tests for scripts. Logged errors will result in validation errors.

3.2.6 Toolbar (2)



Information about **Deploy**, **Patch**, **Commit, Revert** and **History** are available in the manual *Lasernet Configuration Server and Deployment (Lasernet Config and Deployment)*.

Add, **Edit** and **Remove** are used to maintain the objects for Servers, Modules, Forms, SubForms, Phrases, Scripts, Database Commands, Modifiers and Profiles objects.

Add	Select an object category and click Add button to create a new object in the selected object category.
Edit	Select any object in any category and click the Edit button to edit the properties of the object.
Remove	Select any object in any category and click the Remove button to delete the object from the entire configuration.

3.2.7 Tool buttons (4)

The contents of the property bar vary depending on which tools you have in focus.

Modules

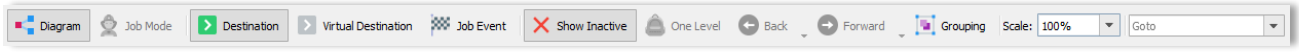
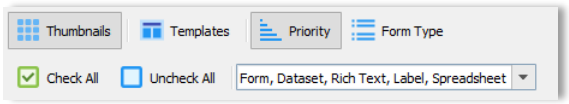


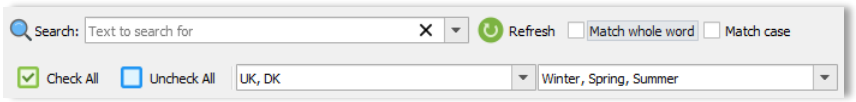
Diagram view is an alternative to the list view. You can toggle between list and diagram views via the tool button above the modules. More information regarding Diagram mode is available in *Chapter 1.9.1 Diagram Mode.*

Forms



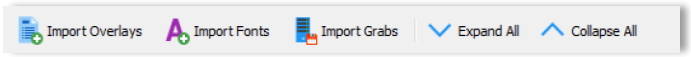
The Thumbnails view is an alternative to the list view. You can toggle between list and thumbnails view via the tool button above the forms.

Phrases



A search tool to filter the list of phrases by typing a search string or defined list of keywords that match your phrases. More information about phrases is available from the Lasernet Form Editor Guide in the Formpipe Knowledge Base.

Resources

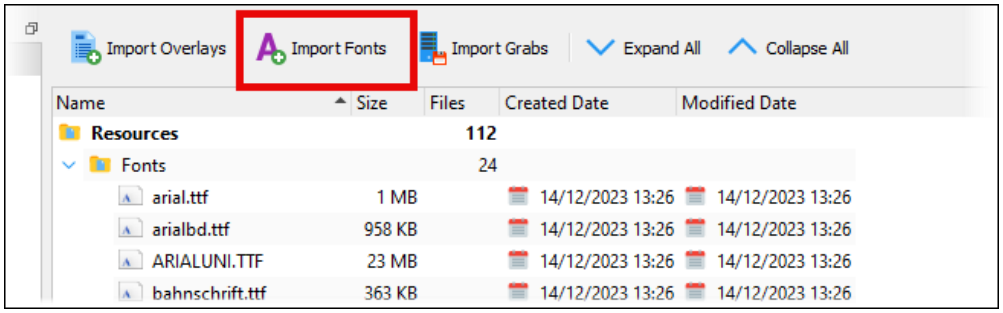


Click **Import Overlays**, **Import Fonts**, or **Import Grabs** to store those items as resources in the configuration. Expand and collapse the folders in the **Resources** area of Lasernet Developer to explore the resources stored by the configuration.

Import Fonts identifies which Windows fonts are used by the forms and phrases in the configuration and then copies them to the **Fonts** folder of the configuration's **Resources** area. Fonts that are stored within the configuration can always be accessed by the Lasernet Form Editor, Lasernet Developer, and by Lasernet servers, when working with EMF-based output such as PDF.

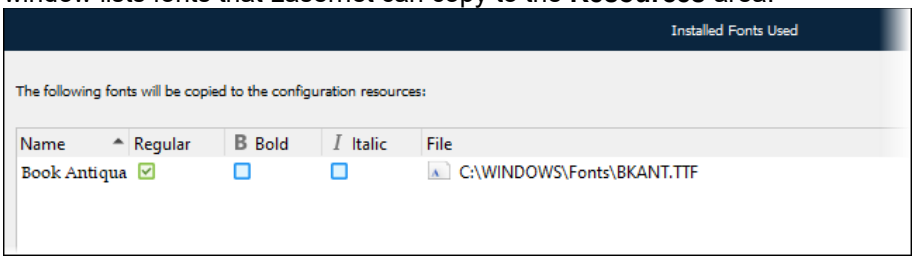
The benefits of storing fonts in the configuration are that the fonts are easily shared between users of Lasernet Developer and there is no requirement to manually install fonts on Lasernet servers.

Click **Import Fonts** to add the used fonts to the configuration.

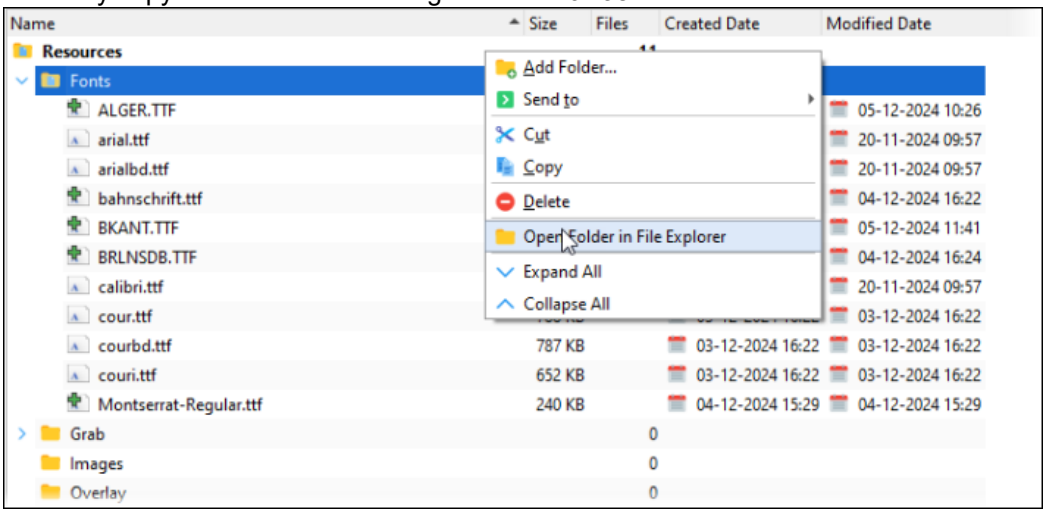


Note: Other methods for adding fonts to the configuration as resources are:

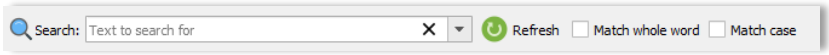
- When the Form Editor saves a form, it identifies any fonts that are used in the form design but are not present in the **Fonts** folder of the configuration's **Resources** area. An **Installed Fonts Used** window lists fonts that Lasernet can copy to the **Resources** area.



- Manually copy a font file to the configuration's `Fonts` folder on disk.

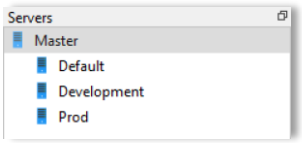


Scripts and Commands



A search tool to filter the list of scripts and SQL commands by typing a search string that matches your contents.

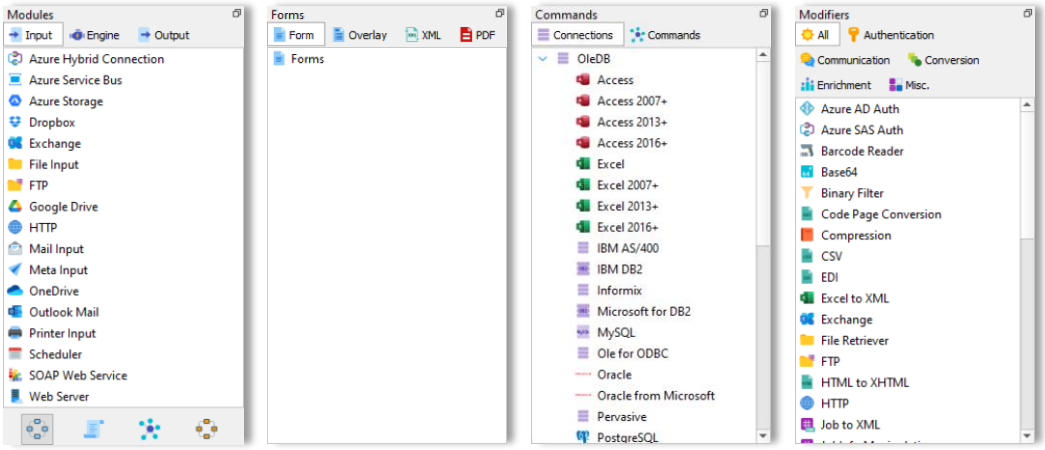
3.2.8 Servers (4)



Choose the Server tool to show the list of servers and select which one you want to configure. Select **Master** to assign and set your objects to be inherited by all the servers in the list or select a specific server to assign your objects for that server only.

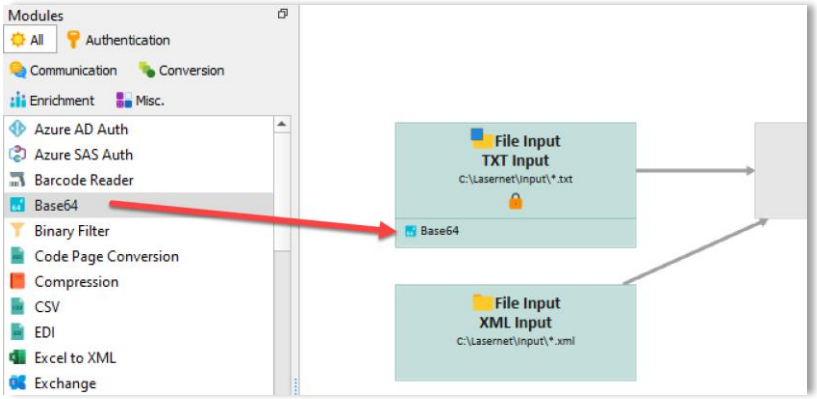
3.2.9 Objects (5)

The contents of the objects vary depending on which tools you have in focus.



The object contains all available object types you can configure in the workflow. Double click or drag them into the main window in diagram or list view mode.

In module mode you will also have shortcuts for adding script, command and modifier objects to your workflow by dragging them to a module in the main window.



3.3 Export/Import of Objects

Exporting and importing single objects between configurations is only recommended between the same major versions of Lasernet. This is due to potential configuration formats changing between versions. Otherwise, you must import a full configuration, which will migrate objects to the Lasernet 10 format, before an export and import of a single object is recommended. Note: As an experimental feature, Lasernet Developer does not prevent the import of objects in an older format, however results are not guaranteed.

Note: Printer Profiles are not objects and, therefore, cannot be seen in the object list when using "File - Export Objects". However, Printer Profiles are linked to the Printer Output objects for which they were

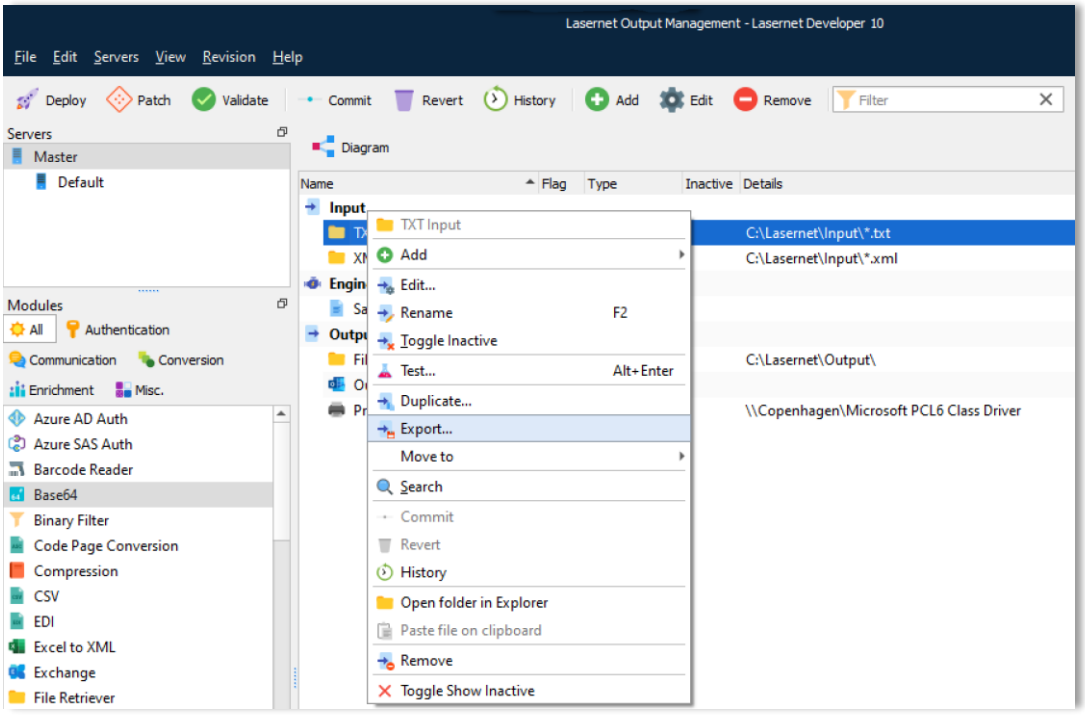
created. When you export a Printer Output object, any associated Printer Profiles will be included with that object, making them available when you import the printer into another configuration.

You cannot select individual Printer Profiles under the printer object when exporting. All associated Printer Profiles will be included in the **.lnobjectx** export file.

3.3.1 Export

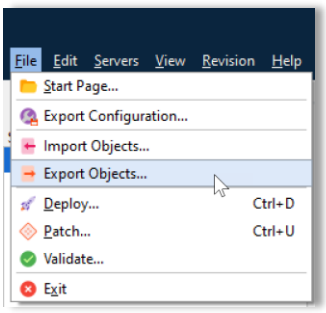
There are two ways of exporting objects from a Lasernet configuration.

1. Use the context popup menu when you right click an object.

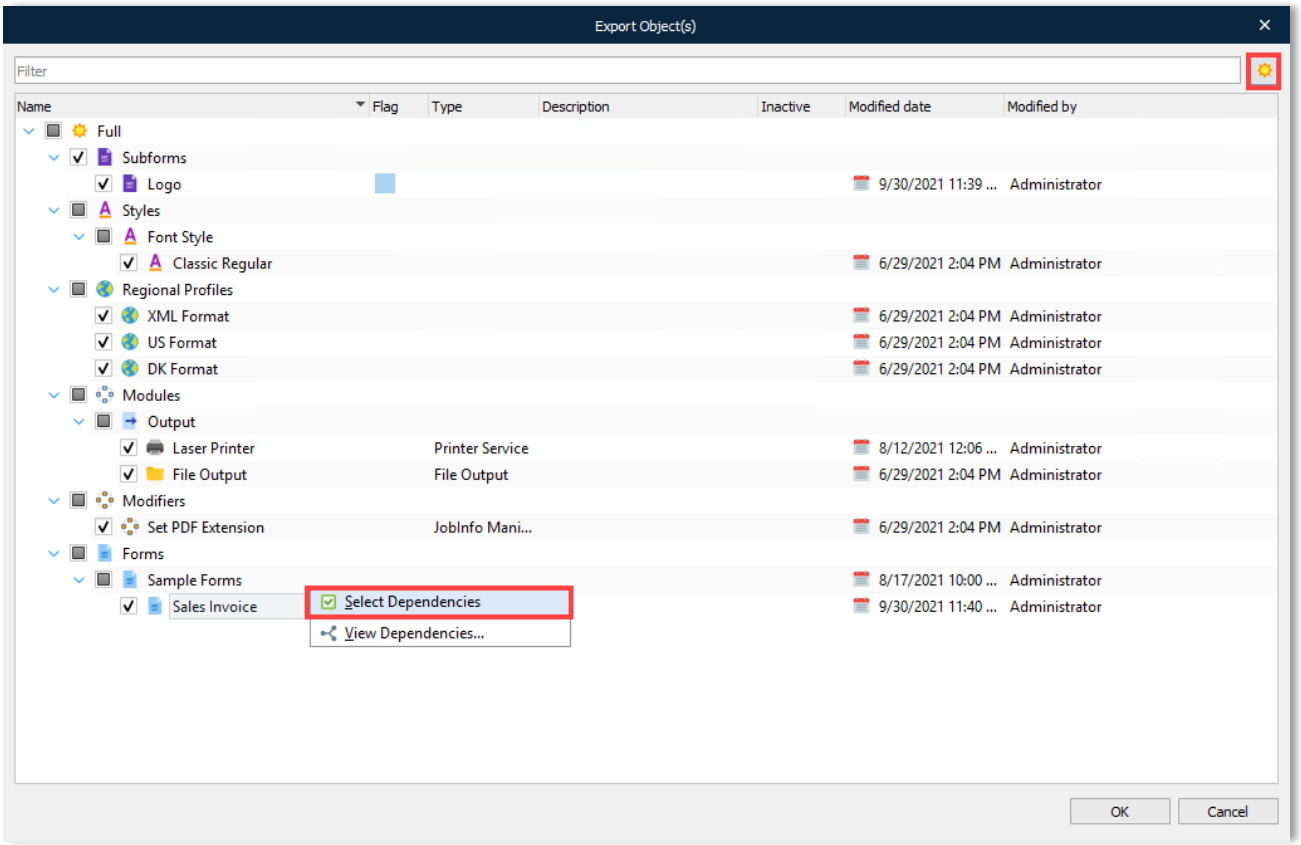


Lasernet will then prompt you for a folder to export to. Here a '.settings' file will be saved along with any accompanying files.

2. From the **File** main menu, multiple objects of any type can be exported.



A dialog is displayed where one or more objects can be selected.



You can select any type of forms and modules of the following by clicking the relevant checkbox:

- A full configuration
- A group of objects
- An individual object

Click the  icon to toggle between showing either all objects, or just the selected objects.

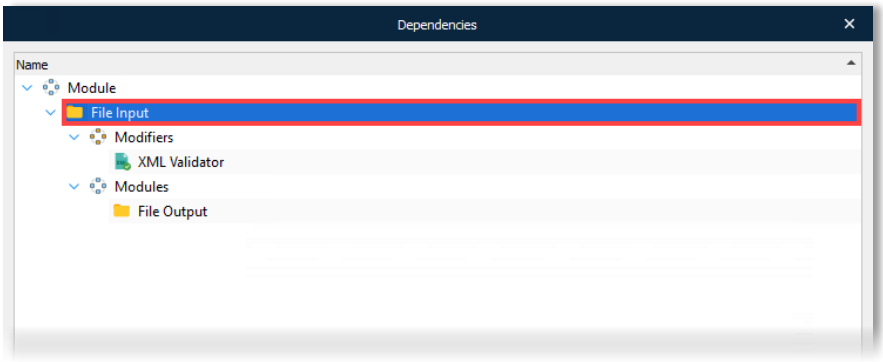
Dependency Walker
You can right-click an individual object to display the following options:

Select Dependencies – Lasernet will automatically select all dependent objects.

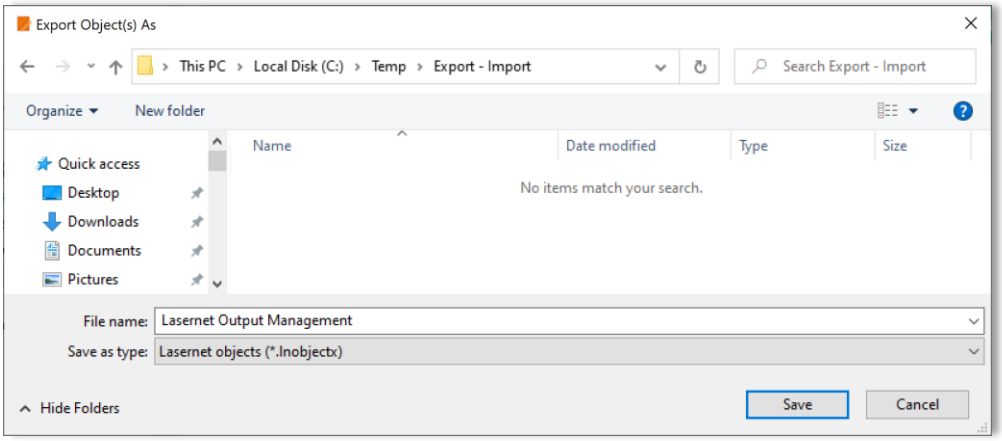
Note: Lasernet will not show:

- More than one level of dependencies
- References used by script functions
- References from JobInfo Substitution
- DataType profiles used from Web Server input
- Database Connections added to SharePoint module

View Dependencies – Displays all dependencies to the selected object.

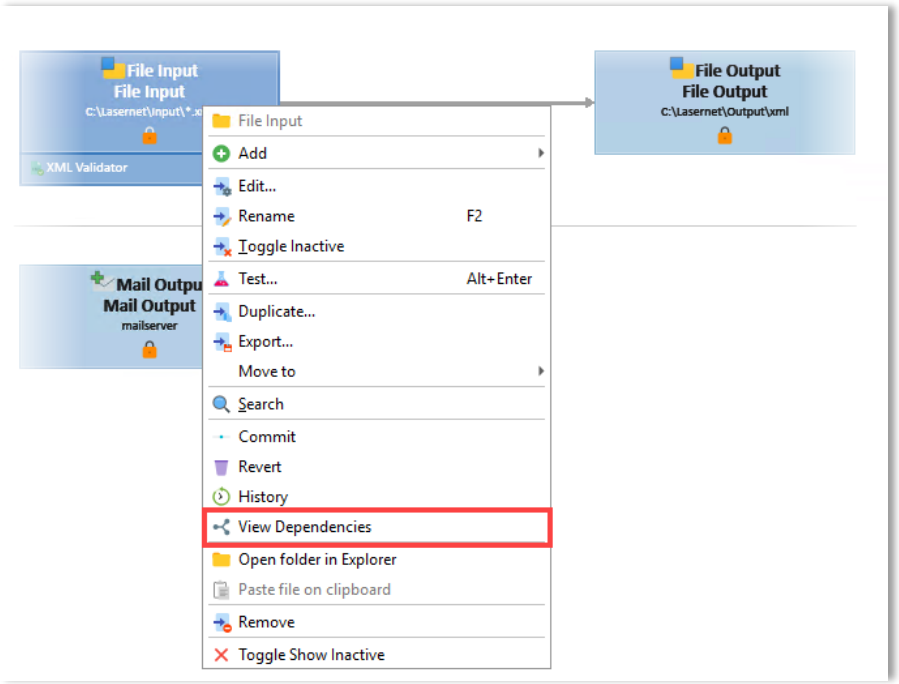


Click **OK** and a prompt will ask for a folder to export to.



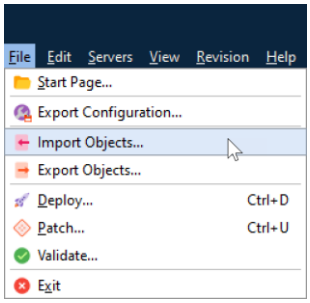
Select a drive and a folder. Type a new file name or select the name of the configuration (default value). The extension of an exported package is .lnobjectx.

Note: You can also select **View Dependencies** by right-clicking a module or form.

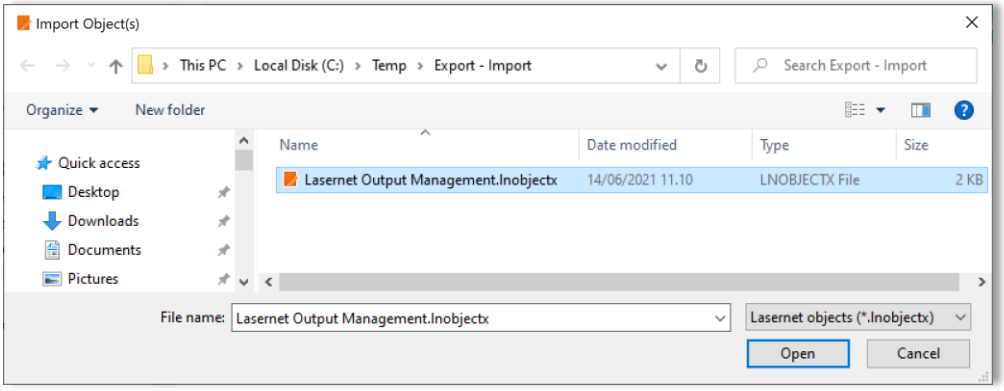

File Input
File Input
XML Validator
File Output
File Output
C:\Lasernet\Output\xml
Mail Output
mailserver
File Input
Add
Edit...
Rename F2
Toggle Inactive
Test... Alt+Enter
Duplicate...
Export...
Move to
Search
Commit
Revert
History
View Dependencies
Open folder in Explorer
Paste file on clipboard
Remove
Toggle Show Inactive

3.3.2 Import

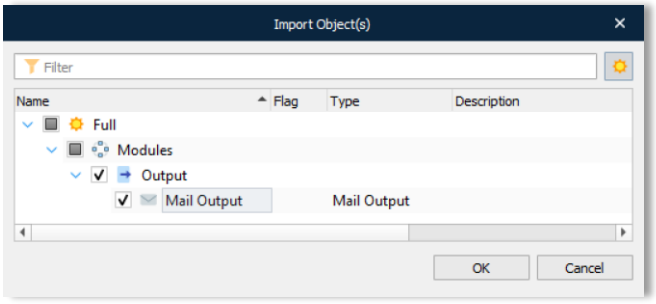
Importing is done from the **File** main menu.



A file browser dialog is displayed where one or more exported '.settings' files can be selected.



When the folder has been chosen you can select all available objects (Full) that are found in the folder, or you can select specific objects for your import.



While importing, a progress screen is displayed where the current object can be seen.

It is important to note:

1. Imported objects are linked to the 'Master' server.
2. If the object name already exists a prefix "_1" will be added to the object name.
3. Modified fields are cleared and created fields are updated with current date, time and user.

4. When importing a Database Connection from another setup, please note that the setting for the Connection Name will be lost and a “Deleted” text will appear instead. You must manually assign it to an existing Database Connection in your setup. This is because the connection is not imported with the original GUID and is assigned a new one instead (meaning the Command refers to an unknown connection).

Export and Import of Database Commands
Caution: A Database Command contains within it a reference to a Database Connection. After exporting and importing Database Commands between different configurations, the value for the Database Connection will be marked as “DELETED!”, as a valid GUID to the Database Command no longer exists.

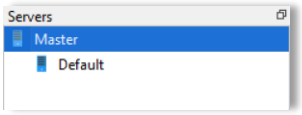
To fix this, enter the properties for the Database Command and manually choose one of the available Database Connections listed in the configuration.

4 Servers.

4.1 Servers

4.1.1 General

The name of a server has to be added in the configuration setup. Use the name of the computer where the Lasernet 10 service is installed.



Server names

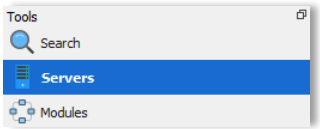
When entering server names, only a specific set of characters are allowed. The set is described in RFC 952 (http://www.faqs.org/rfcs/rfc952.html): (A-Z), digits (0-9), minus sign (-), and period (.) plus one addition: the underscore (_) character, which is commonly, used in NETBIOS names. Other special characters such as: ! @ # $ % ^ & ') (– { } ~ [space] are not allowed in computer names in Lasernet. The reason for not allowing these characters is the transition to Active Directory in Windows 2003 which uses the DNS-naming scheme, which prohibits use of these characters. NETBIOS support in Windows 2003 exists only for backwards compatibility. *Please note that it is not possible to use an IP-address as computer name.*

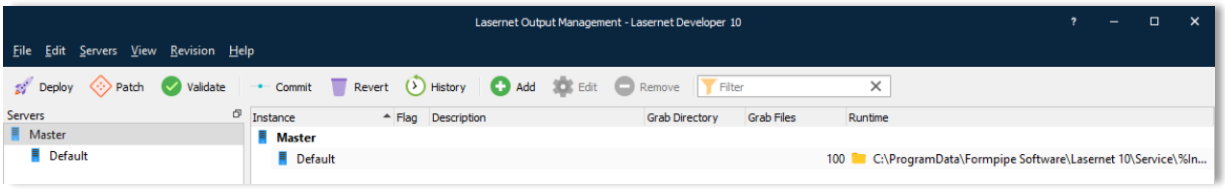
Master server

When you create a new configuration, Lasernet Developer will automatically create a Master server for you. This means that all modules etc., that are added for this computer will be inherited by all other computers in the configuration. This makes it easier to share settings between multiple machines by placing only those components that need to differ under the corresponding server.

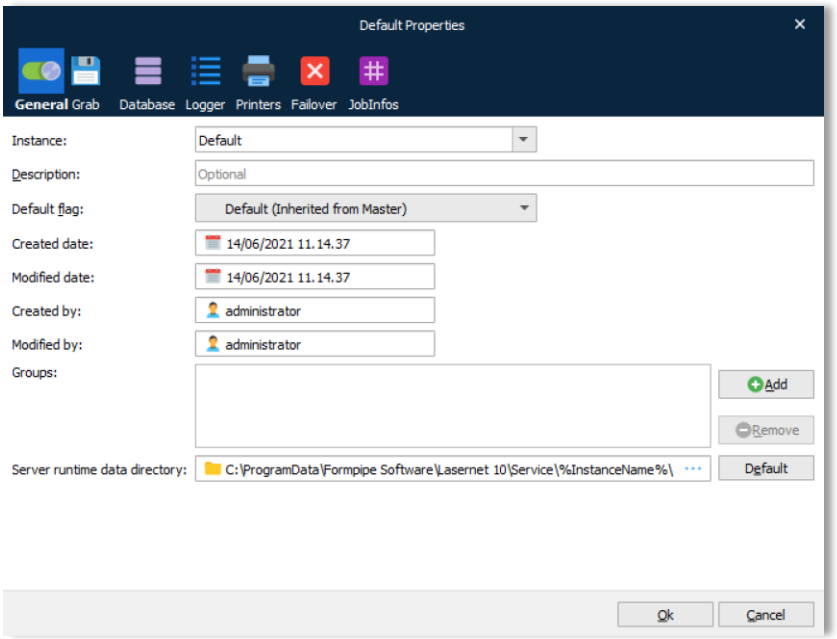
Managing server

Choose the Server tool to show the list of servers.





Then press the **Add** tool button to add a new server.

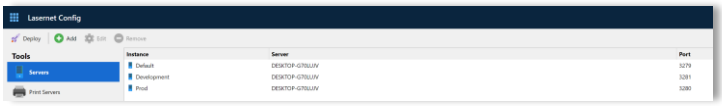


Press **Ok** to confirm the ”Server name” or **Cancel**, if not accepted.

To edit the properties of a Server Setup, right-click the Server Name and in the drop-down menu select command “**Edit**...”

Instance

The instance name is linked to the server name and port number set on the Lasernet Config 10 server. Click the dropdown button to select an existing instance name.

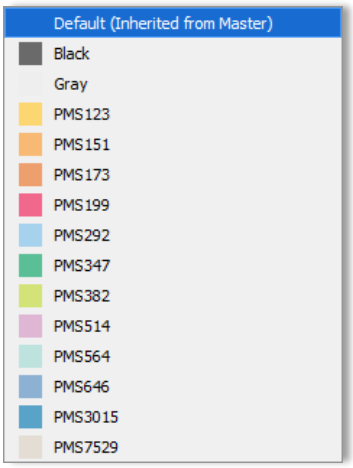


Description

An optional field used to describe the server and settings.

Default Color

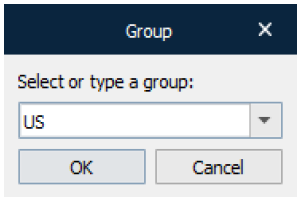
This parameter is only used for the visualization of modules in the Lasernet Developer. The default background color setting is used when drawing/listing added modules in diagram or view list mode. The default color is a light gray, but can be set to any color available in the pre-defined list.



A color can be defined for each server to make it easier to see which modules are connected to which server.

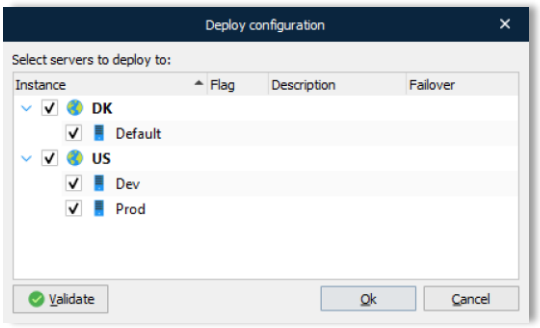
Groups

A server can belong to a user defined group or none. Click Add to connect the server to a group or Remove if you want to exclude the server from an added group.



Select an already existing group in the configuration or type the name of a new group.

Grouping names are used when updating a configuration. Instead of selecting a specific server name, a group of servers can be selected.



Server runtime data directory

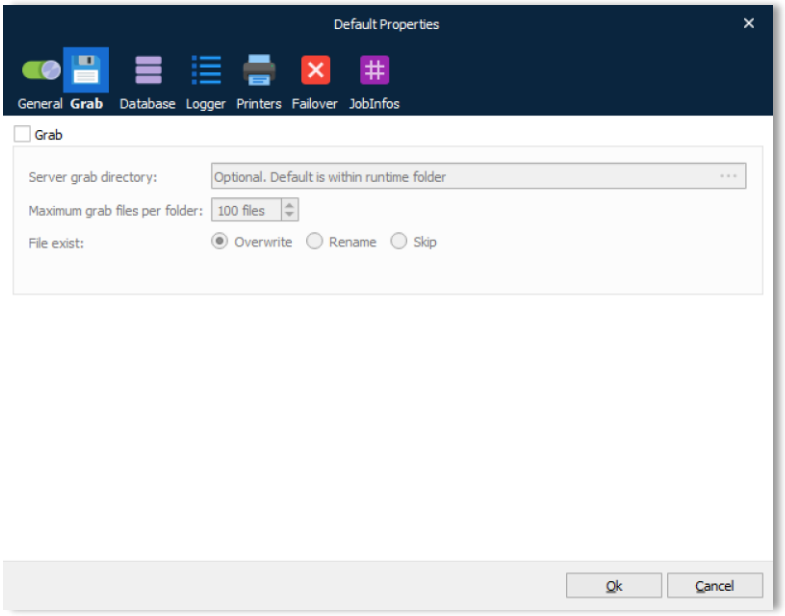
Location to store temporary files, failed jobs, scheduled jobs and combined jobs. This folder is created on the server,

which can be different from the machine where Lasernet Developer is started.

4.1.2 Grab

Grab mode is used by the Lasernet Server to store a copy of the job data in a user defined server grab directory. Grab files are required by the Form Editor as input data for designing forms or the XML Transformer Editor to modify XML data.

4.1.2.1 Server settings



Enable grab mode

Select this checkbox to activate the grab mode for forms and modules where grab mode turned is on. Lasernet will not enter grab mode, for any form or module before the configuration is uploaded to the server. The Lasernet service will always start up in grab mode, after a restart of a Lasernet service, or when this setting is activated.

In the general settings for modules and forms, you must also define what event level the Lasernet service should create grab files, before the grab mode will take place.

Enabling/disabling Grab mode at module and form level can also be managed from the Grab page in the Lasernet Monitor. This is convenient if you want to remotely activate/de-activate grab mode, without updating the configuration.

Server grab directory

The root directory for grab data. A sub directory will be created in the root directory containing the name of module or form name where grab mode was activated. The Server grab directory will automatically be created on the Lasernet Server during processing of job data.

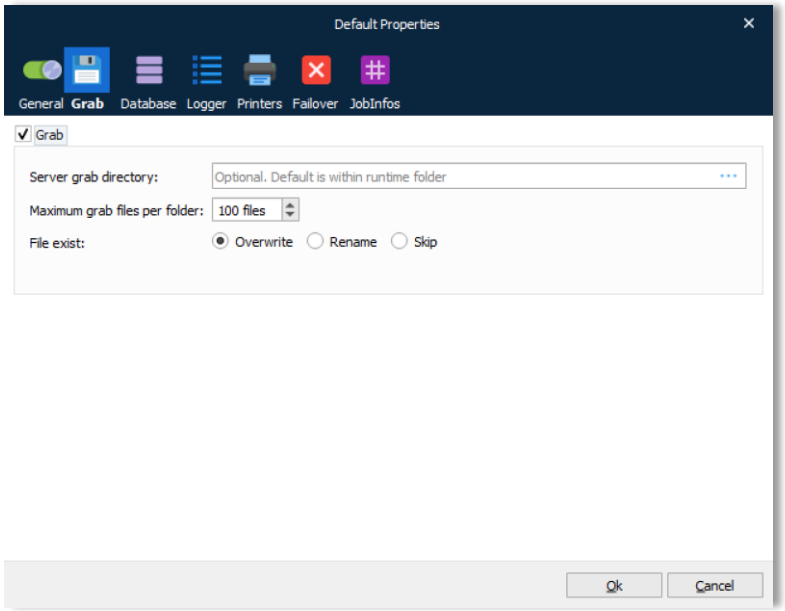
If no directory is specified grab files will be stored in the Server runtime data directory specified in the General tab settings.

Maximum grab files per folder

Maximum number of grab files per folder, before the oldest grab files are automatically deleted.

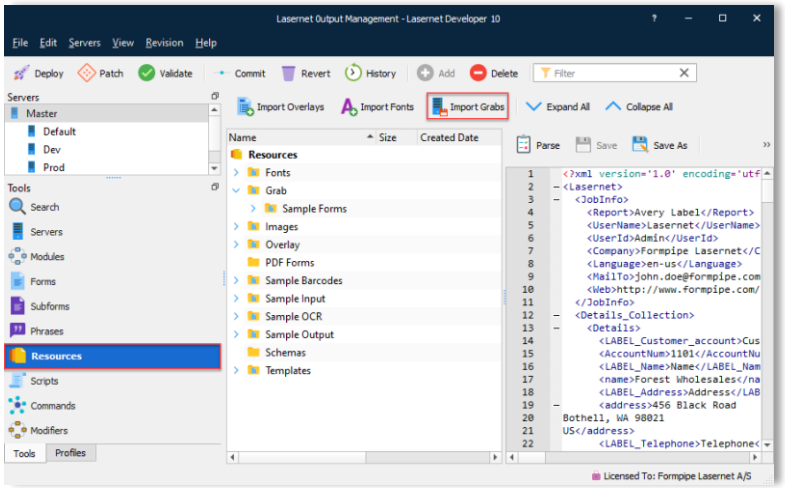
File Exist

Set behaviour if the file name already exists in the folder. Select overwrite, rename or skip. Rename will enable additional settings where you can define a splitter symbol and the number of digits in the filename.



Download grabs from server

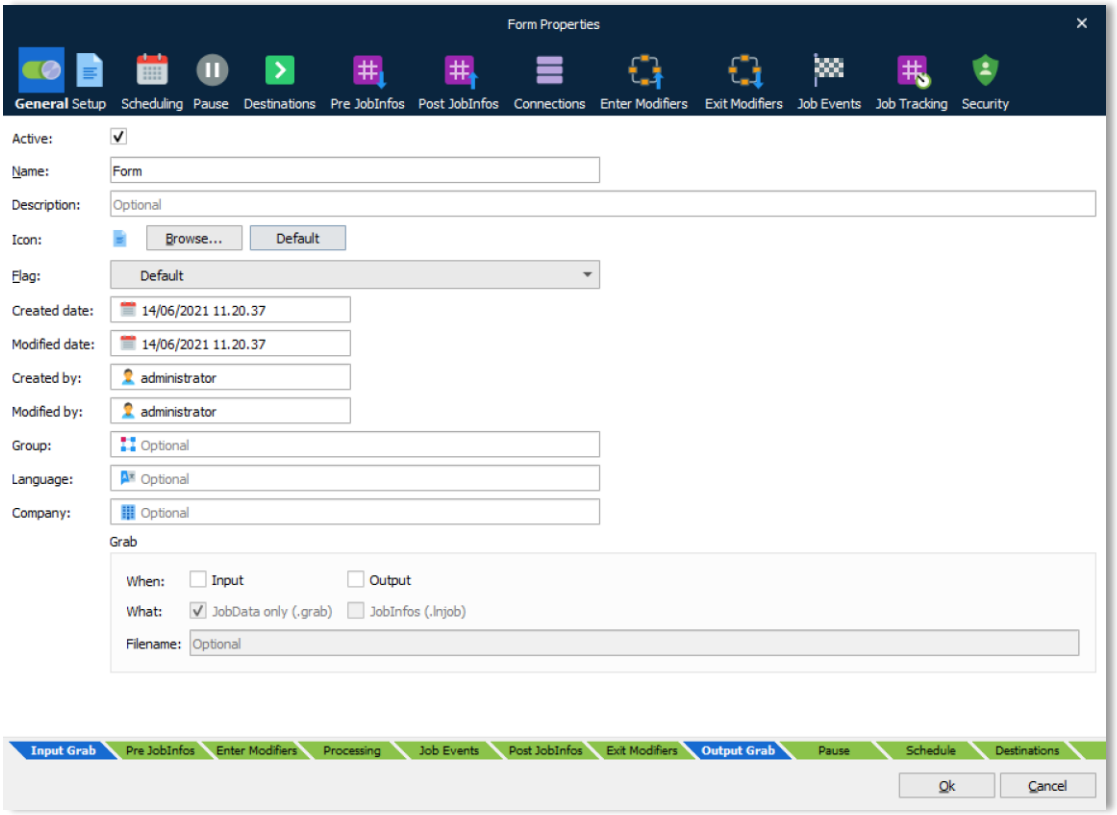
Go to Resources and select the Import Grabs button to import all stored grab files from the selected server.



Grab files will be transferred from the server grab directory to the destination of the Lasernet configuration file. After being transferred, downloaded grab files will be removed from server.

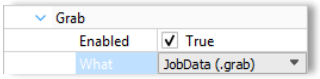
4.1.2.2 Activate grab mode at module and form level

Grab mode can be activated on any module added to the configuration. In the module you can define if the grab mode should run before data enters the module (input) or when data exits the module (output). Grab mode has to be activated in the server settings first, before enabling grab mode on a module will have any effect.



To help understand what event grab data will be stored, the tab bar at the bottom of the general settings shows the order that data is processed in the specific modules. Input grab mode is always performed before JobInfos and Enter Modifiers are executed, and output grab is always performed after Exit Modifiers are run. This can have an effect on what the contents of the grab file is.

If you only need to grab specific forms, grab mode can also be activated and de-activated at form level. To set grab mode for multiple forms, enable the grab mode on the property editor and select what to grab.



When adding new modules, the ‘enable’ setting is deactivated by default. To activate grab mode at module level you have to activate both the grab mode in the server settings and in the module settings.

In the form object, the ‘enable’ setting is activated by default when adding new forms. The is also true for XML forms added to the XML Transformer Engine.

We recommend that you keep these default settings as they will ensure that Lasernet 10 behaves in the same way as previous versions. Grabbing data at form level is the most common way of grabbing data in Lasernet and is essential when working with data design in the Form Editor or XML Transformer Editor.

Enabling grab mode at module level is often only used to analyze data or to create a backup of a processed job at that level, where it is most convenient to re-use the job in a future test.

4.1.2.3 The hierarchy of grab folders

Grab files are stored in a hierarchical folder structure off the top-level folder defined in the server settings. When grab files are downloaded from the server, the folder structure is retained in the configuration directory.

In the following example, grab modes have been activated for the forms, “Avery Label”, “Postings”, “Product Sheet”, “Sales Confirmation” and “Sales Invoice”. On the module “XML Input”, grab mode has been activated at the input event.

In the grab folder, the grab files will be stored in a relative folder containing the name of the form or the name of the module + grab event (Input/Output). This will help you locate the grab data when browsing for grab files.

Grab /ERP Forms /Avery Label

/Postings

/Product Sheet

/Sales Confirmation

/Sales Invoice

/XML Input /Input

If you want to setup a more general grab mode that works for all forms, you can disable grab mode on form event and activate grab mode on the Form Engine at the input event. This will grab any job parsed through the Form Engine where grab mode has been activated.

If you want to analyze the output for all forms generated by the Form Engine, you can activate grab mode on output level. If both grab events are activated, the grab files will be saved in the following folder structure:

Grab /ERP Forms /Input

/Output

The same way of activating grab mode will also work for the Overlay Engine and XML Transformer Engine, both of which can contain a list of forms.

4.1.2.4 Types of grab files

You can adjust the “What” setting in order to grab both at module and at form level. This will cause the Lasernet server to save one or two grab files for each job.

JobData (.grab)	A file which contains only the content of the JobInfo ‘JobData’. No JobInfos are available.
JobInfos (.lnjob)	A file which contains the full Job including all JobInfos.

4.1.2.5 Activate grab mode from Lasernet Monitor

You can remotely activate any module or form added to the configuration from the Lasernet Monitor.

More information about how to activate grab mode from the Lasernet Monitor is available in the manual Lasernet 10 - Monitor.

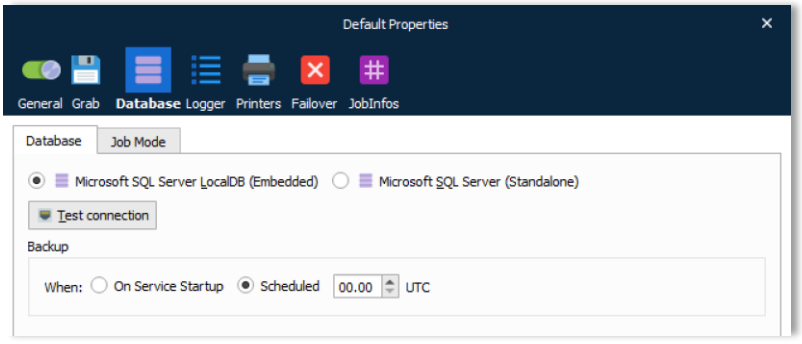
4.1.2.6 Work with grab data in the Lasernet Form Editor

More information about how to work with grab data in the Lasernet Form Editor is available from the Lasernet Form Editor Guide in the Formpipe Knowledge Base.

4.1.3 Database

Each job has a selection of system created JobInfos (metadata), which are stored in the database during the processing of a job. When a job has successfully been processed, the metadata is deleted and no references for the job are available afterwards.

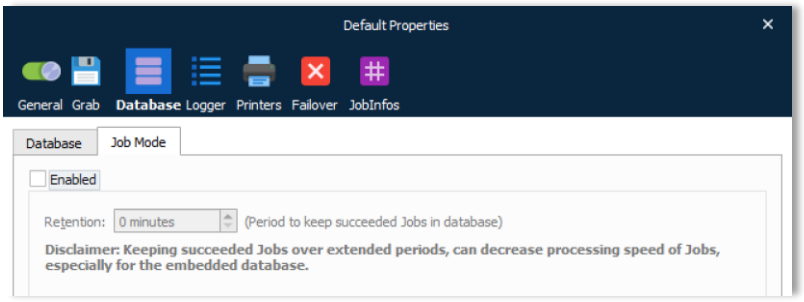
Paused, scheduled and failed jobs are stored temporarily in the database until they are released and processed successfully. When jobs are in temporary storage they can be viewed and managed from the Lasernet Client or they can be tracked the jobs and view the job flow in Diagram → Job Mode.



As an alternative to the embedded Microsoft SQL Server LocalDB database, you can connect to a either a Microsoft SQL Server Express or Microsoft SQL Server, for storing jobs.

4.1.3.1 Job Mode

If Job Mode is not activated (default), jobs are not stored in the database for a specified retention time.



This is the fastest way to process jobs through Lasernet. If the value for Retention is set to 0 (zero) minutes, information about the job is stored in the database as long as the job is running, but deleted immediately after it is successfully processed. If you want to keep JobInfos in the database for successfully processed jobs, you must set a positive Retention value (in minutes). Only JobInfos (metadata) will be saved, not the physical job (JobData). Archiving must be enabled to save a copy of the JobData (more information can be found later in this chapter).

When the retention is reached for a completed job, metadata will automatically be deleted from the database. Job retention has no effect for paused, scheduled and failed jobs. They will not be removed from the database, before they are deleted manually or released from the Lasernet Client and processed successfully. More info about how to view, release or delete jobs can be found in the manual for the Lasernet Client.

Disclaimer: Keeping completed jobs over extended periods, in the embedded database, can decrease the processing speed of jobs.

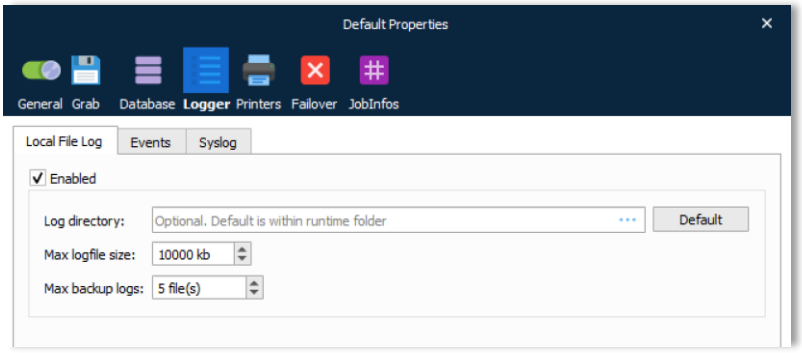
4.1.3.2 JobInfo Profiles and Security Roles

More information can be found in the JobInfo Profiles and Security Roles chapter, regarding how to set up configuration rules for the paused, scheduled, failed and succeeded jobs. JobInfo profiles are required to specify the metadata selection to be saved with each job. Security Roles are required to specify which user/group is allowed to view and manage saved jobs in the database from the Lasernet Client.

4.1.4 Logger

During processing, the Lasernet service will generate log messages which can be used to determine processing status and forms flow within the Lasernet server.

Log data will be sent to the Lasernet Monitor, which is a useful administrative tool for viewing the live log. Read the “Lasernet 10 - Monitor” documentation for more information on how to use the Lasernet Monitor.



4.1.4.1 Local File Log

Log data can also be sent to log files, in a user defined directory, containing the same information as sent to the Lasernet Monitor. The log files will contain historical information about processes and forms flow, which can later be opened from the Lasernet Monitor.

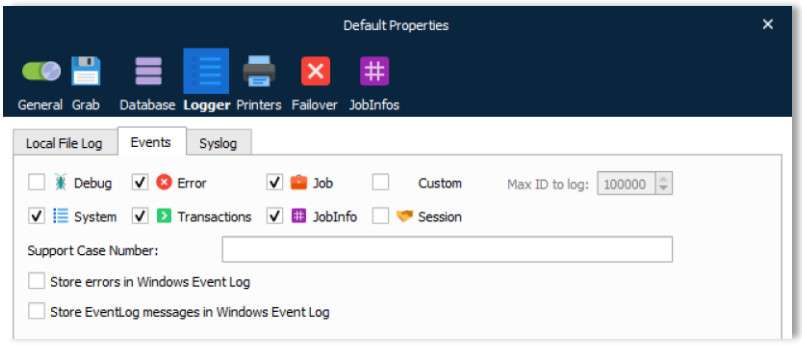
Log directory	The directory in which to save the log file.
Max log file size	Maximum file size for the log file.
Max backup logs	Maximum amount of log files.
Enabled	Turn on/off logging to local file log.

We recommend that you use a local drive as a log directory for faster processing. In the chosen directory, the file named Lasernet.lnlog contains the newest log information. When the log file has reached the max log file size, it will be renamed to Lasernet.1.lnlog and Lasernet.1.lnlog will be renamed to Lasernet.2.lnlog etc.

The default settings cap the maximum log file size to 10,000KB. With five (5) full backup logs, this will result in a total of 50,000KB of log data.

4.1.4.2 Log these events

Each system defined log line is associated with a log event type. You can configure the event type you want to save in the log file. By default, Debug mode is not activated, since this is primarily used for developers to store very detailed information about a processed job.

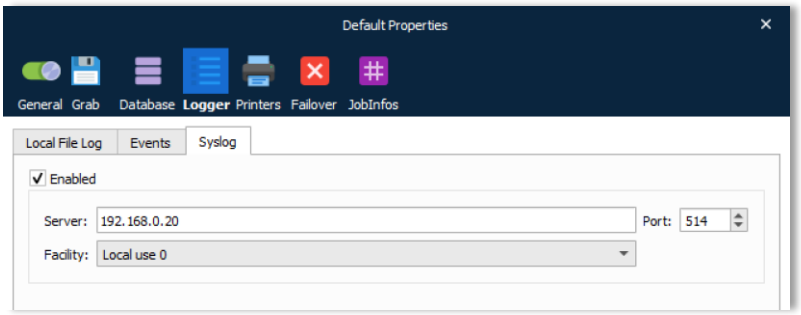


Debug	Events that generally produce too much noise to be included in EID_JOB or EID_SYSTEM, but which may provide extra help tracking down a problem or a possible bug in the system
Error	Fatal errors that prevent a job from being successfully received, handled or sent to an output port.
Job	Log data from the modules as a job passes through them. The Overlay Engine may comment on what overlays are put on or what JobInfo is found. The Forms Engine logs what form was chosen. Information that helps show why the job passes through the system as it does.
System	Boot and shutdown messages. License information about modules.
Transaction	Log information that shows how a job is received by the input port, handled by the engines and committed by output ports. Mostly produced by the Job class and the transaction manager.
JobInfo	Set JobInfo log events.
Session	This option applies to all modules which communicate with other systems to intercept the session log. Sessions are logged for: HTTP, FTP(S), SFTP, POP3, IMAP, File Retriever (FTP(S), SFTP, HTTP), SharePoint and Web Service.
Custom (Max ID to log)	User defined log messages. By setting “MAX ID to log” you can choose to log only event ID’s, lower than the defined number.
Support Case Numbers	This field is for internal use only. Allows an individual or list of support case numbers, separated by a comma, to be entered.

4.1.4.3 Syslog

Server option to activate the ability to log to a remote Syslog server via UDP (as per the RFC5424 standard).

All log events, selected for logging in the Events tab, are forwarded to Syslog server. Syslog UDP packages are limited to 500 bytes, which leave ~480 bytes for the message itself (depending on machine name of the Lasernet server etc.).



Server The name or IP address of the Syslog server.

Facility A facility code is used to specify the type of program that is logging the message. Messages with different facilities may be handled differently. The list of facilities available is defined by RFC 3164.

Facility codes

The mapping between facility code and keyword is not uniform between operating systems and different syslog implementations.

Facility code	Keyword	Description
0	Kernel	Kernel messages
1	User-level	User-level messages
2	Mail	Mail system
3	System daemons	System daemons
4	Security/authorizations	Security/authorization messages
5	Syslogd/internal	Messages generated internally by syslogd
6	Line printer subsystem	Line printer subsystem
7	Network news system	Network news subsystem
8	UUCP subsystem	UUCP subsystem
9	Clock daemon	Clock daemon
10	Security/authorizations	Security/authorization messages
11	FTP daemon	FTP daemon
12	NTP subsystem	NTP subsystem
13	Log audit	Log audit
14	Log alert	Log alert
15	Clock daemon	Scheduling daemon
16	Local use 0	Local use 0 (local0)
17	Local use 1	Local use 1 (local1)

18	Local use 2	Local use 2 (local2)
19	Local use 3	Local use 3 (local3)
20	Local use 4	Local use 4 (local4)
21	Local use 5	Local use 5 (local5)
22	Local use 6	Local use 6 (local6)
23	Local use 7	Local use 7 (local7)

Severity

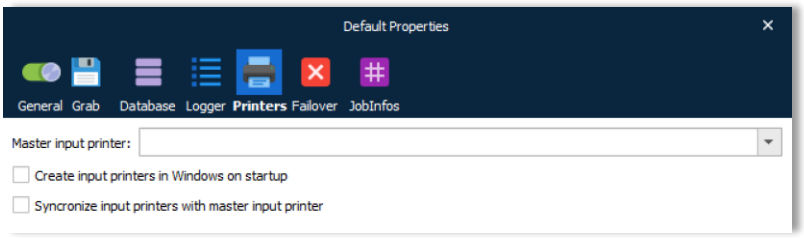
Syslog	EventID	Description
Error	EID_ERROR	Fatal errors that prevent a job from being successfully processed
Warning	EID_WARNING	Warnings
Informational	EID_TRANSACTION EID_SYSLOG EID_CUSTOM	Transaction log for JobEngine Customized logging for Syslog server only Customized logging for Lasernet Server
Debug	EID_DEBUG EID_JOB_PUBLIC EID_JOBLOG	Debug logging Log data from the modules as a job passes
NOT logged	EID_IMMEDIATE EID_EVENT_LOG EID_NOLOG	EventIDs in this category not logged to the Syslog server

Any other EventIDs are logged as Informational.

4.1.5 Printers

Any Printer Input Port added to the Lasernet configuration, will be added as a Lasernet Printer Port in the Windows Spooler System by default, when updating a configuration to the Lasernet server.

The Lasernet server is also able to automatically create the actual printer queues in Windows and connect them to the respective printer port if a “Master input printer” is defined and “Create input printers in Windows on startup” is activated in the Master Printer tab.



Master input printer — The name of the Windows printer which is used as the master printer, when creating or synchronizing input printers. Printer queue settings defined for the master printer will be copied to new or synchronized input printers.

Create input printers in Windows on startup — Any Lasernet printer input port defined in the configuration, will automatically be created as a Windows printer queue during service startup.

Synchronize input printers with the master input printer	Any Windows printer queue created by Lasernet, will automatically be synchronized with the settings saved in the master input printer, during service startup.

This feature is used to ensure that any Lasernet input printer added to the configuration, is always available on the Windows printer server. If a user makes breaking changes to the Windows printer queues by mistake, Lasernet is able to synchronize and reconfigure the settings, as defined for the master input printer.

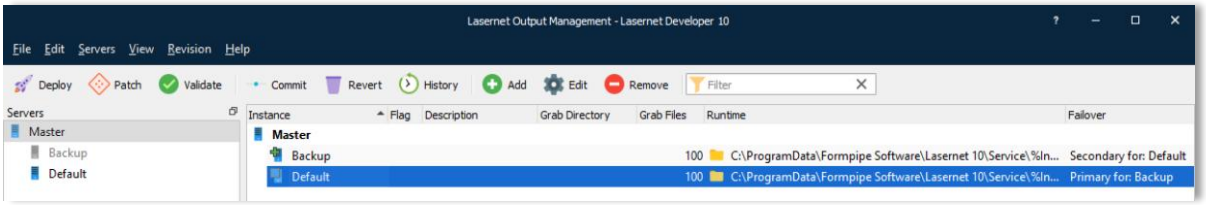
This feature is also useful if Lasernet contains a huge list of printer input ports. You only have to manually create the master input printer in the Windows printer environment, with the required settings for the type of printer driver, paper format, share settings etc. During service startup, Lasernet will then automatically create a Windows printer queue for any of your listed printer input ports in the configuration and apply the configured settings from the master printer.

Please note that the Device Settings configured in the printer queue settings will not be synchronized as they are not a part of the basic printer device mode settings.

4.1.6 Failover

A failover environment must consist of two physical Lasernet Servers.

- Primary server
- Secondary server (backup server)



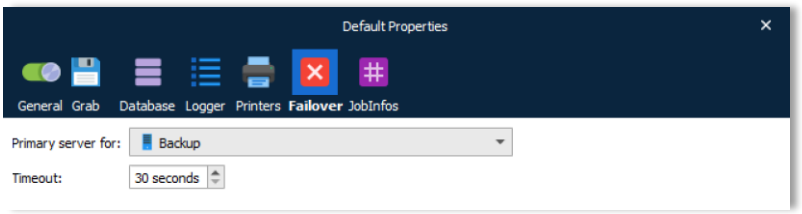
Add two servers to the list and ensure that your firewall settings on the Windows Server will allow the Lasernet services to communicate on the selected port number.

In a failover scenario you must ensure that your input ports are still able to receive data delivered from your clients. It is important to design your configuration and 3rd party connections with this in mind.

Read the section about recommendations for the setting up your database, embedded vs. Microsoft SQL Server, in the section “Database”.

4.1.6.1 Sharing the configuration

Go to the Failover tab and select which server is the secondary server (backup server) for the primary server.



The primary server and secondary will share the same configuration, apart from the setting which determines which server has primacy.

The secondary server will have its input ports in passive mode unless the connection to the primary server is lost. Then the secondary server will activate its input ports until the primary server is ready to accept jobs again.

One or more Lasernet Developer clients can connect to the servers, query them for status and update their configuration.

4.1.6.2 Startup
Both of the servers will always start with their input ports set to passive.

4.1.6.3 Startup of Primary Server
The primary server will send a **CONFIGURATION** to the secondary server and activate its input ports when it receives a response or when the instruction times out. If the instruction times out, new attempts will be made at sending the configuration until a response is received.

4.1.6.4 Startup of Secondary Server
The secondary server will send a **GETCONFIGURATION** to the primary server. If that instruction times out the secondary server will activate its input ports. Otherwise, it will resume normal operation.

4.1.6.5 Normal Operation
The primary server is responsible for regularly sending the **ACTIVE** heartbeat to the secondary server. The secondary server will listen for heartbeats from the primary server. As soon as no heartbeat is received during a timeout period, the secondary server will assume that the primary server is not operating, and activate its input ports, until an **ACTIVE** or **CONFIGURATION** is received from the primary server.

4.1.6.6 Abortive Primary Restart
The failover system is primarily designed for handling the unintentional loss of the primary server. When the primary system experiences a power loss or network failure, it is classified as an abortive primary stop. Whenever the system is restarted, it must synchronize with the secondary server to get back control over incoming jobs.

4.1.6.7 Graceful Primary Restart
When there is a need to stop the primary server e.g., when the computer must be restarted, the primary server can communicate this to the secondary server, which then activates its input ports without the need to wait for the time-out period.

4.1.6.8 Secondary Restart
When the secondary server (backup server) needs to stop, nothing needs to be communicated to the primary server. Therefore, there is no difference in an abortive or graceful secondary restart. When the secondary server is restarted, it asks for the latest configuration.

4.1.6.9 Updating Roles Setup
Roles are updated from the client (Lasernet Developer). When a client wants to set up a pair of computers, a full configuration is sent to the primary server, together with the address of the secondary server. The client

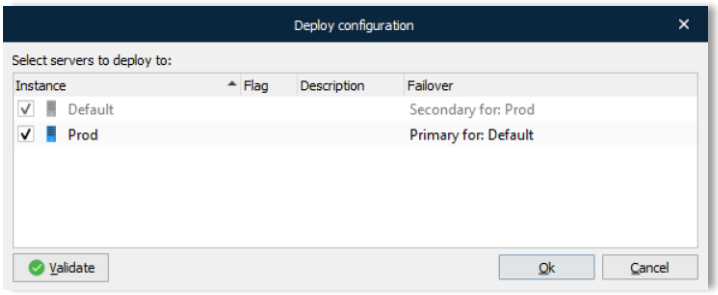
also sends a message to the secondary server instructing it to become a secondary server and which primary server to listen to.

This ensures robustness if more than one primary server accidentally ends up sharing a secondary server, and instructs the secondary server to sync to it.

If a secondary server receives a **SETASSECONDARY** instruction with an address different from the current, it will immediately perform a **GETCONFIGURATION** on that new server.

4.1.6.10 Updating Configuration

The primary and secondary server will synchronize their settings without the need of client interaction. That makes it possible for the Lasernet Developer to just update the primary server's configuration and the servers will make sure the change is transferred to the secondary server. The update from the client can be made even if the secondary server is down.

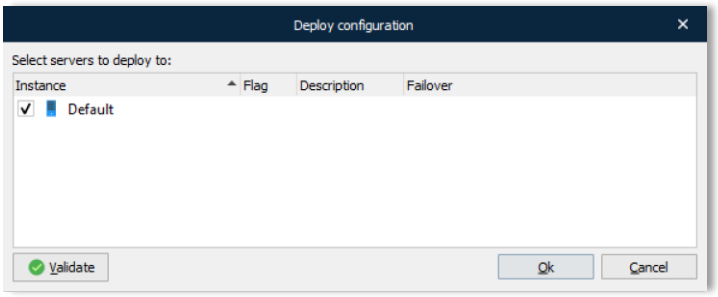


When the primary server configuration is updated, the primary server will send the **CONFIGURATION** to the secondary server.

5 Updating the Lasernet Server.

5.1 Deploy a configuration to the Lasernet Server

In the Lasernet Developer select **File -> Deploy** to deploy a configuration.



This gives you the ability to select the servers to deploy to. If you have uncommitted objects or resources, a warning will appear. You can now decide to deploy with uncommitted objects/resources or you can select the Commit button and commit your changes before the deployment. Click **Ok** and the transfer will start deploying the configuration to the selected Lasernet 10 service(s). A dialog shows the progress of the update.

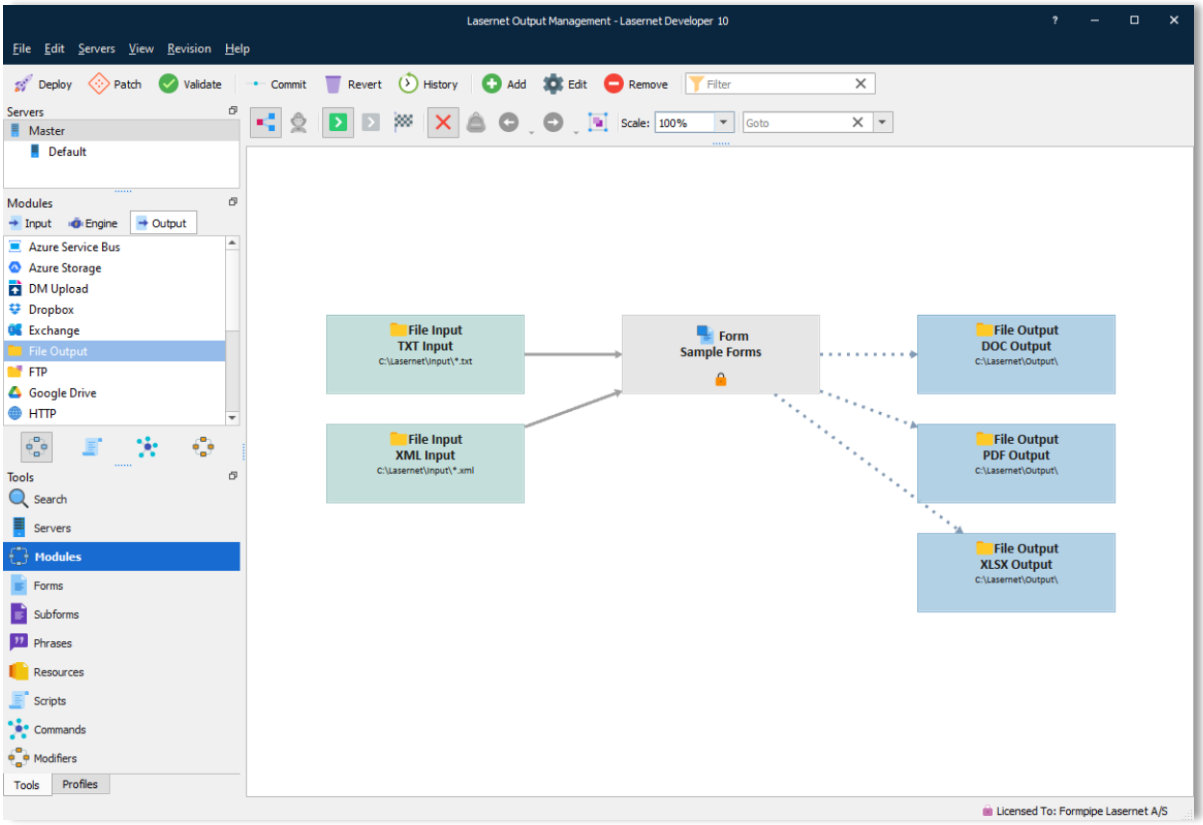
After deployment, the Lasernet 10 service will automatically reload the new configuration and the Lasernet Configuration Server will create a new revision of the configuration.

You can start the Lasernet Monitor to see whether the update succeeds. For details, see document “Lasernet 10 - Monitor”.

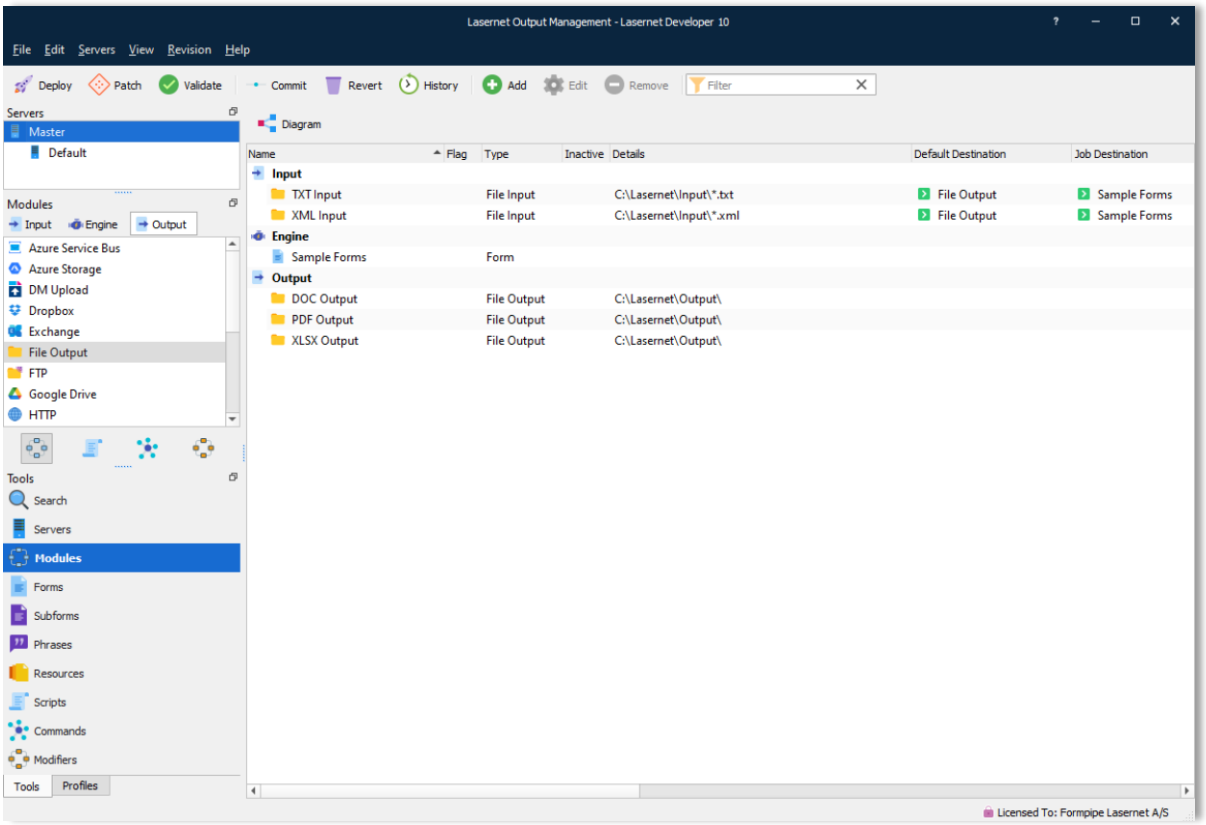
6 Modules and Engines.

6.1 Modules

This is where all input modules, engines and output modules are managed. The server bar shows which server has been selected as the active server.

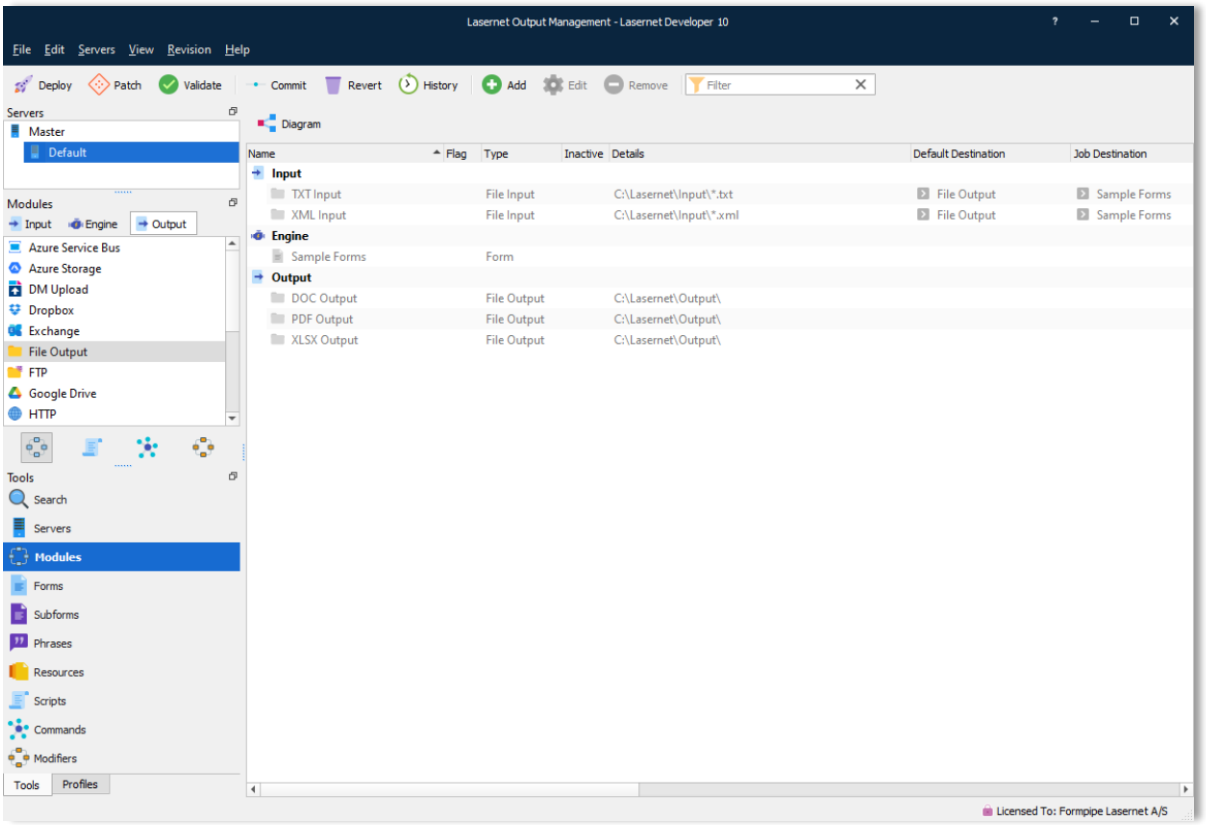


You can view and maintain the modules in two modes. By clicking the Diagram button, you can switch between Diagram mode (Default) and List mode. In both modes you are able to manage all modules and maintain their settings, either by double-clicking on the module or by setting the properties from the property editor.



The *Master* server is the template server.

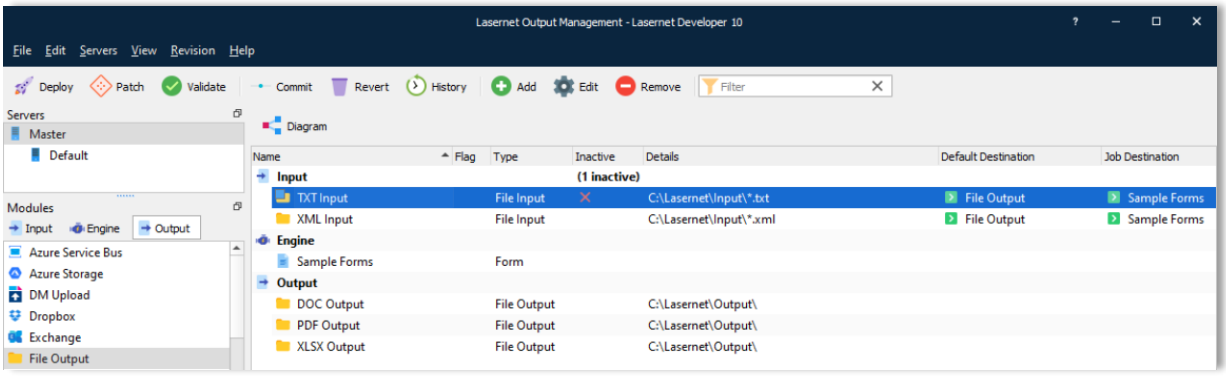
In this particular setup, all modules exist for Master server and will be inherited by any server added to the list.



When changing to another server, some modules for *Master* are greyed out. Modules which are not greyed out will be available for that specific server only and cannot be inherited by any other servers listed.

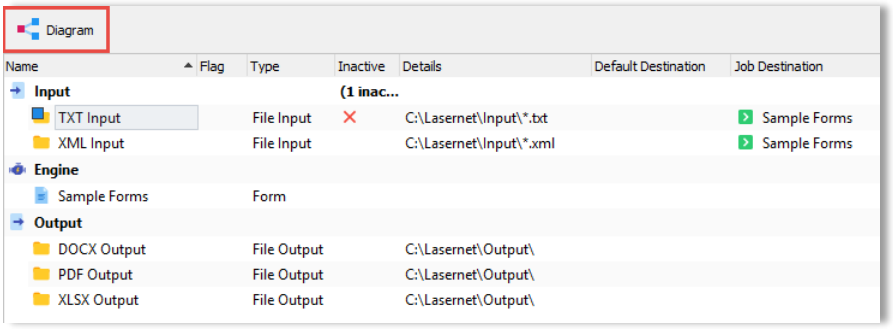
6.1.1 Inactive Modules

Modules can be marked as *inactive*. This means that they will not be started when uploading the configuration to a Lasernet Server. You can toggle between the active/inactive status by right-clicking the modules and choosing **Toggle Inactive**. If a module is inactive, it is marked by a red cross in the column **Inactive**.

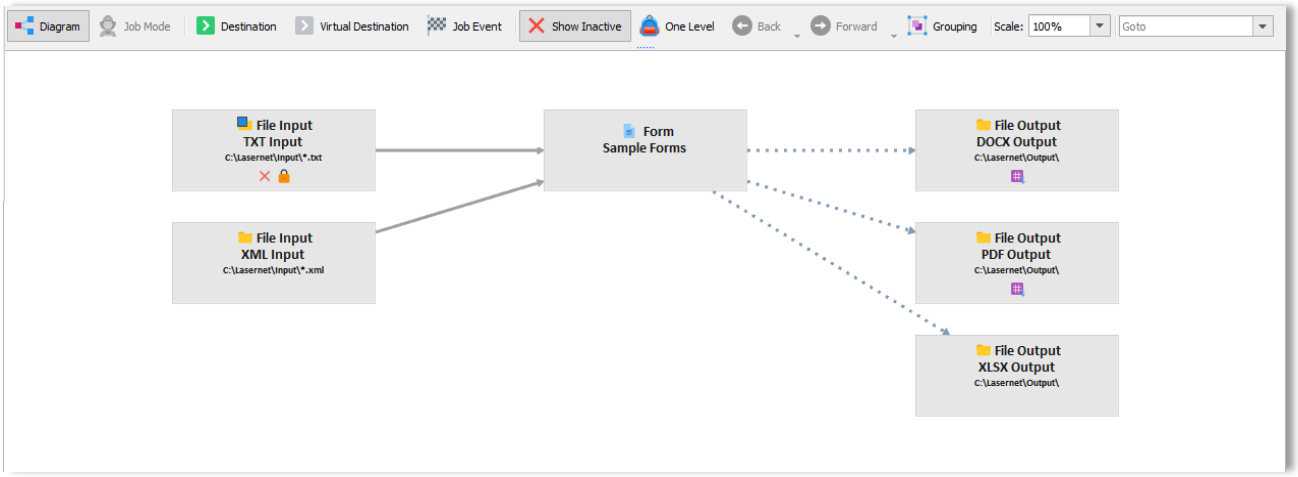


6.1.2 Diagram Mode

Diagram mode is an alternative to the list view. You can toggle between views using the tool button above the modules.



When Diagram is toggled on, you are presented with a top down overview of your configuration. This makes it much easier to see how Jobs are passed between the modules in the configuration.



In Diagram mode you can create and design the configuration in the same way as in the list view.

New modules can be added by: dragging them from the left side of the screen onto the diagram, by using the Add button or using the popup menu.

Modules can be edited by: using the property editor, double clicking the module, choosing edit from the tool bar or using the popup menu.

Deletion of selected object(s) is done by: pressing the delete key, using the delete tool button or choosing delete from the popup menu.

Destination / Job Event

Create new destinations or Job Events by dragging a line between two modules. Properties for the destination / Job Event are shown in the property editor and it is possible to double click the arrows to add or change criteria.

When a criterion is added to a destination or Job Event, the arrow line is dashed.

Virtual Destination

Destinations created via script commands and inserted on sheets in forms, will not be visible as destination lines in the diagram view. To help create a more useful / accurate diagram showing which objects are actually linked together, you can drag virtual lines between modules. The virtual line has no function when a job is parsed, it is only used as a guide to help to visualize your configuration.

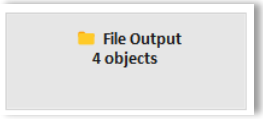
When virtual destinations are used, the arrow line is dotted.

One Level

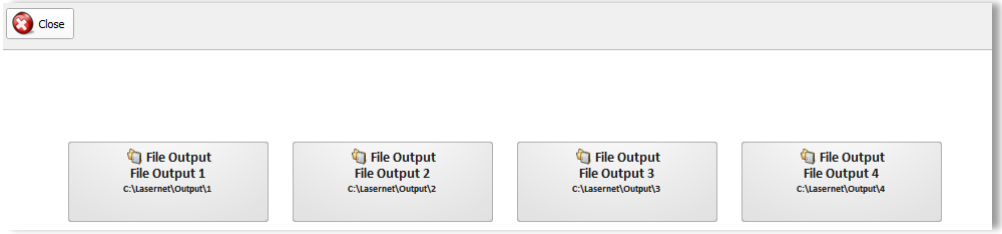
The One Level Browsing feature enables the user to only view modules ‘one level away’ from the selected module or destination/Job Event. The selected module is indicated with a yellow background and blue border. The selected destination/Job Event is indicated in blue. One Level browsing makes it possible to navigate through a configuration, in the same way web pages are navigated in a browser. Back and forward is supported, both via the mouse buttons and the Back and Forward tool buttons.

Grouping

Grouping is a way to arrange similar modules together in order to get a cleaner overview. For example, if you have a lot of Printer Input modules which all go to the same destination, you could group these into a single box.

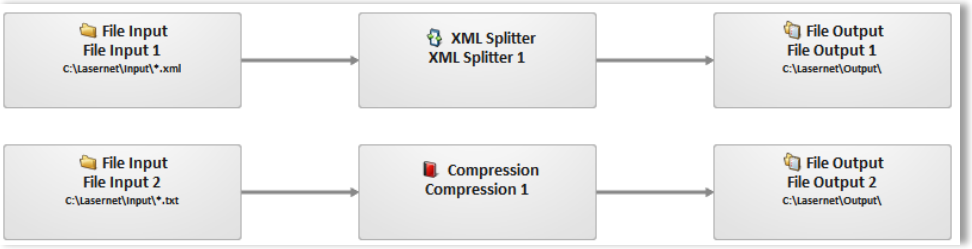


By double clicking the group, you can then access the group view where all of the objects can be managed.



Module selection

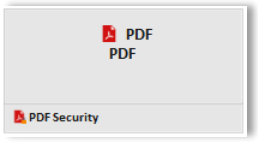
The drop-down box on the right displays a list of modules in the configuration. They can be selected by clicking or by typing one of their names. The module is then highlighted and scrolled in to view.



Modifiers

Modifiers can be run on the enter or exit side of modules. Enter modifiers are listed above the box and exit modifiers are listed below.

The example below shows PDF encryption running as an exit modifier (below module):



Job Mode

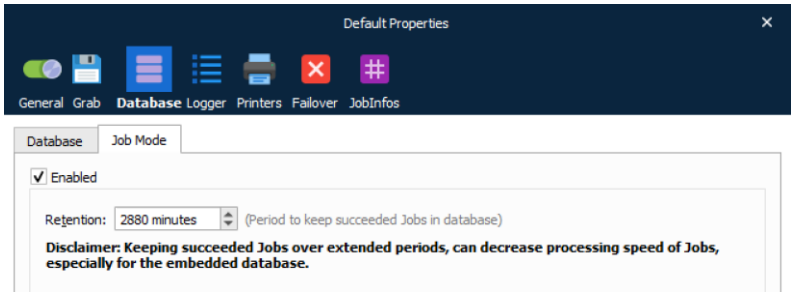
Job Mode allows you to see which jobs have been processed. Jobs are either displayed as green or red, with red indicating that the Job has failed.

Enabling

Before Job Mode can be used a few things need to be enabled:

1. **Job Retention**. By default, all the information regarding successfully processed Jobs is deleted from the Lasernet database. To be able to view this information going forward, Job Mode must be turned on and a value must be set (in minutes) to determine how long Jobs should be stored, after they have succeeded.

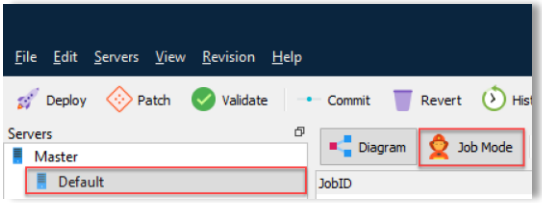
This is done via the server configuration on the Database → Job Mode tab.



2. You must Deploy or Patch the server object to activate Job Mode and reconnect the Lasernet Developer by signing out and signing in again.

Please note: Failed, Paused and Scheduled Jobs are kept “forever”, regardless of this setting.

Usage
Now that Job Mode has been enabled, it can be used on a per server basis in our configuration. By selecting a server, it is now possible to switch to Job Mode.



When entering Job Mode, you are presented by a list of all the processed Input Jobs on the selected server. Initially this list is empty.

6.1.2.1 Mode
There are two modes for viewing Jobs;

5. **Input**: Viewing the Jobs going into Lasernet.
6. **Output:** Viewing the Jobs coming out of Lasernet.

Search
By default, all jobs are listed. However, it is possible to search for JobInfos stored via JobInfo Profiles.

Example
The example above shows two input modules which use different methods of processing in Lasernet.

7. **File Input 1**. Picks up XML files and splits them in to fragments. Fragments are saved to disk by **File Output 1**.
8. **File Input 2**. Picks up TXT files and combines and compresses them into a single ZIP file which is saved by **File Output 2**.

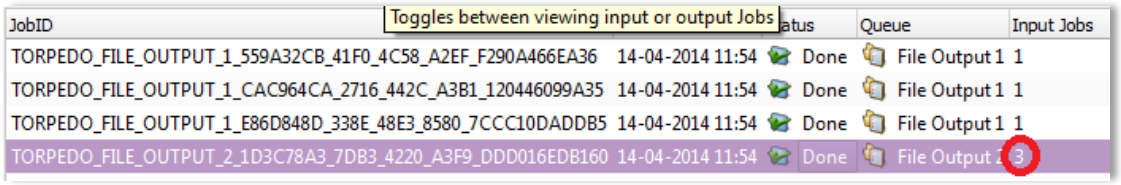
Processing a single XML file which is then split into 3 parts and compressing 3 text files into one ZIP file gives us the following view of the Input Jobs:

JobID	Date	Status	Queue	Output Jobs
TORPEDO_FILE_INPUT_1_AD08BCCD_C58D_451C_BD29_AB7A45C85C80	14-04-2014 11:54	Done	File Input 1	3
TORPEDO_FILE_INPUT_2_B0F1F0B7_5856_4201_BD35_031AD6AE2E4D	14-04-2014 11:54	Done	File Input 2	1
TORPEDO_FILE_INPUT_2_E19CA856_A3C3_442B_8EE8_86C6225A6A6B	14-04-2014 11:54	Done	File Input 2	1
TORPEDO_FILE_INPUT_2_FFB31C06_F1EC_48A9_A4FE_5162E2A40FE1	14-04-2014 11:54	Done	File Input 2	1

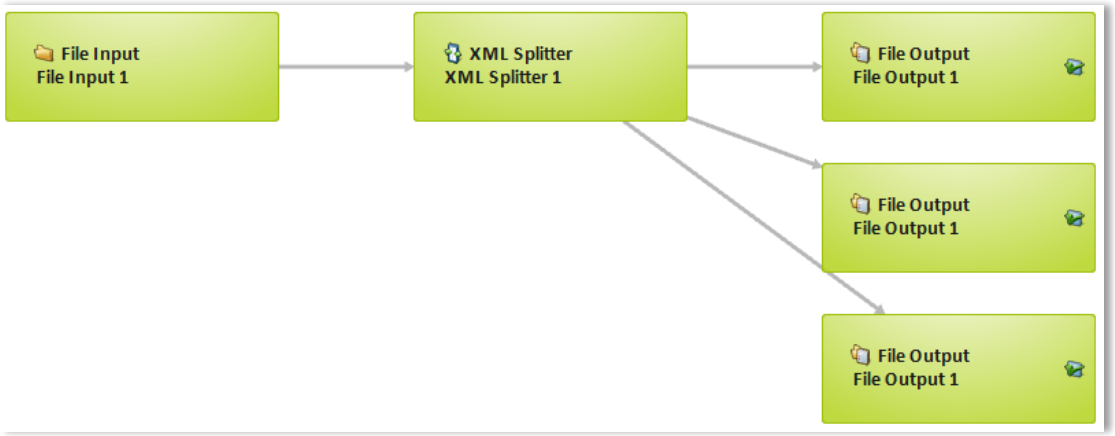
Our one XML file resulted in 3 output Jobs.

Our three TXT files resulted in 1 output Job.

And the output Jobs for these Input Jobs:

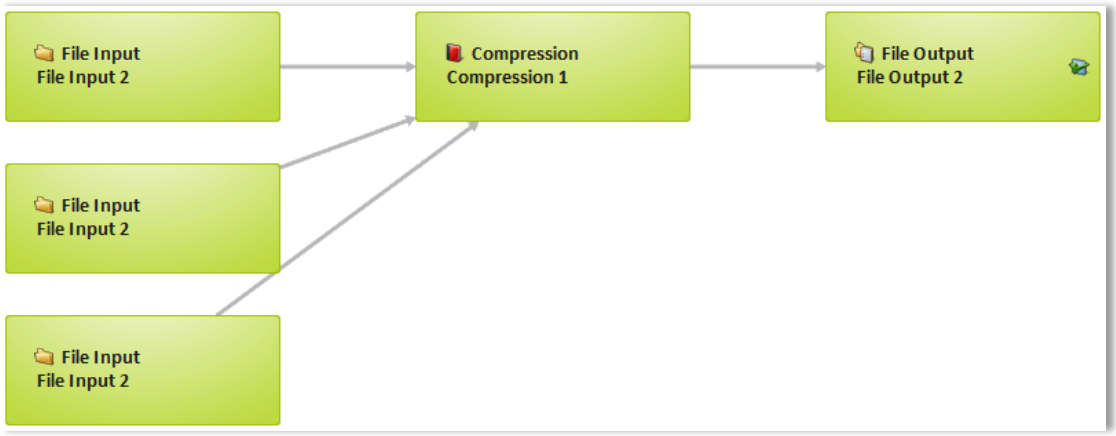


Opening the Job shows us the XML Input processing:



You can see one file coming in and being split into three files coming out.

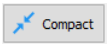
Below you can see how the ZIP file Output Job is processed:



You can see three files coming in and being combined into one zipped output file.

Compact

When Jobs are processed in Lasernet, a number of ancillary Jobs are also created, which can make the diagram harder to follow. To improve readability, you can use the compact button to remove these jobs from the diagram view.

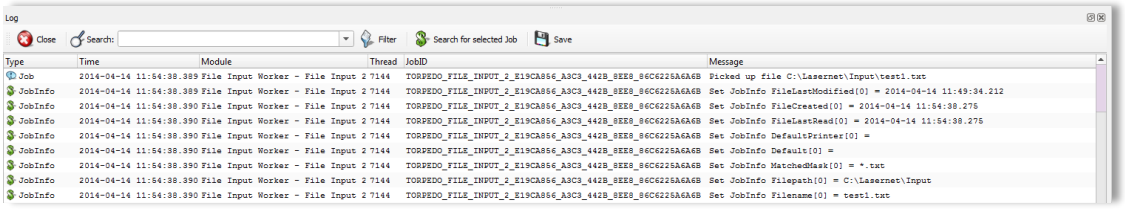


If you prefer to view all of the available Jobs created during processing, you can toggle the Compact feature to ‘off’. This will close the log window (if open) and data will be fetched from the server again.

JobInfos
JobInfos which have been saved to database via JobInfo profiles, are displayed in a property browser along with those JobInfos that are always available, such as Status, JobID and failure reason.

Property	Value
Status	Done
JobID	TORPEDO_FILE_OUTPUT_2_1D3C78A3_7DB3_4220_A3F9_DDD016EDB160

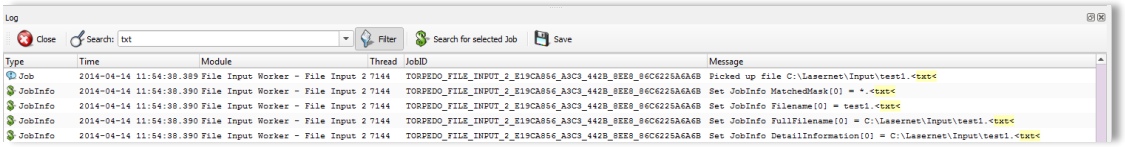
Log
The log lines regarding the viewed Jobs can be fetched from the server by clicking the Log tool button.



A few helpful features make it easier to navigate the log lines.

9. Selecting Jobs highlight the corresponding lines in the log in yellow.
10. The search box enables the user to highlight specific keywords in the log.
11. The filter button toggles between showing all log lines or just the ones being highlighted.
12. Selecting a log line will select the corresponding Job, so the properties are displayed.

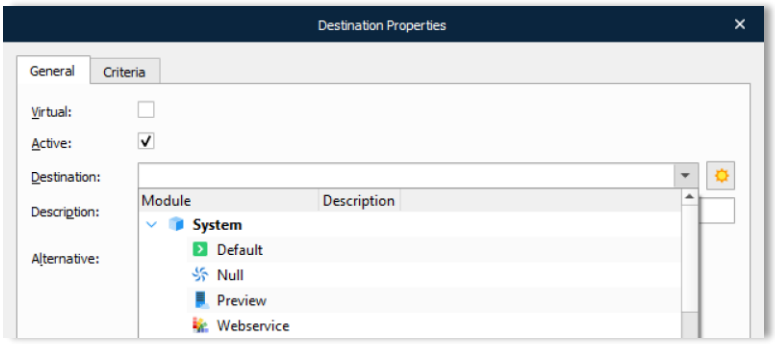
Example of filtering on keyword ‘txt’:



6.1.3 System Destinations

A list of predefined system destinations is available in Lasernet. When listing available **Destinations** for a module, the system destinations will appear in the **System** section. The names of the system destinations are: **Default, Meta** (only listed if a Meta Input module is added to the configuration), **Null, Preview**, and **Webservice**. Lasernet will hide a system destination, if a required setting in the configuration is missing.

To force Lasernet Developer to list all available system destinations, click the **All** icon. 



Default

In the **Destination** tab of an input module, you can define a **Default destination**:

When a job is being processed by the input module, the value defined in the Default destination will be stored in the **JobInfo** named **Default**.

If a module is configured to send a job to the **Default** destination, it will send it to the destination specified in that **Default** JobInfo.

The Default destination enables this module to route jobs to different modules that follow it in the workflow, depending on which input module received the job. This dynamic control is possible because each input can specify a different **Default destination**.

Meta

The Meta destination is only listed if a Meta Input module is added to the configuration. It is used for returning the job initiated by Auto-Fill.

Null

Sending a job to the Null destination will stop the processing of a job in the workflow. If a module does not contain any destinations the job will be marked as failed.

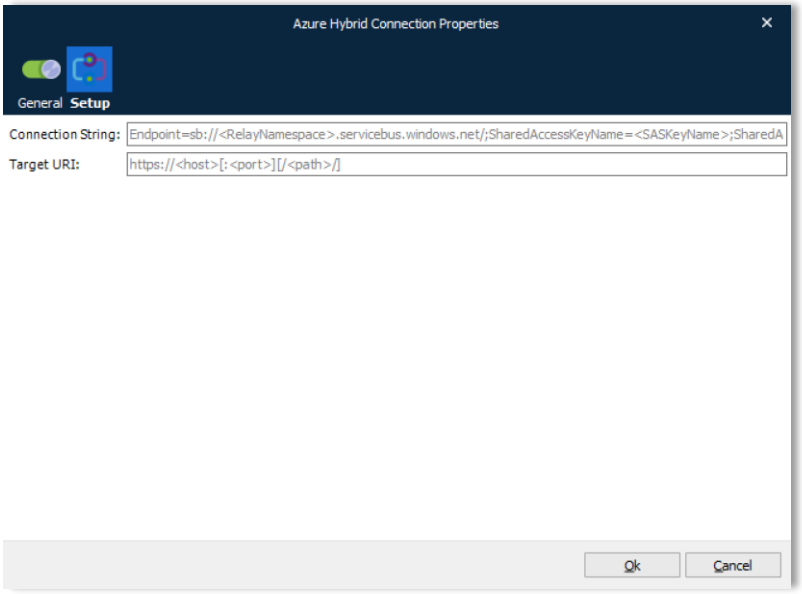
The Null destination is used if you want to ensure that a job is successfully stopped by the Job Engine.

Preview	The Preview destination is added by default if a Web Server module is added to the configuration. More info regarding the Preview destination can be found in the chapter Modules and Engines → Input Modules → Web Server.
Webservice	The Webservice destination is added by default if a Web Server module is added to the configuration. If **Webservice** is selected as a destination, the job must be received by the Web Server module. When the job is received, an open TCP/IP connection is created by the Web Server and the Webservice module will re-direct the job back to the client via the open connection. Afterwards the connection will be closed.

6.2 Input Modules

6.2.1 Azure Hybrid Connection

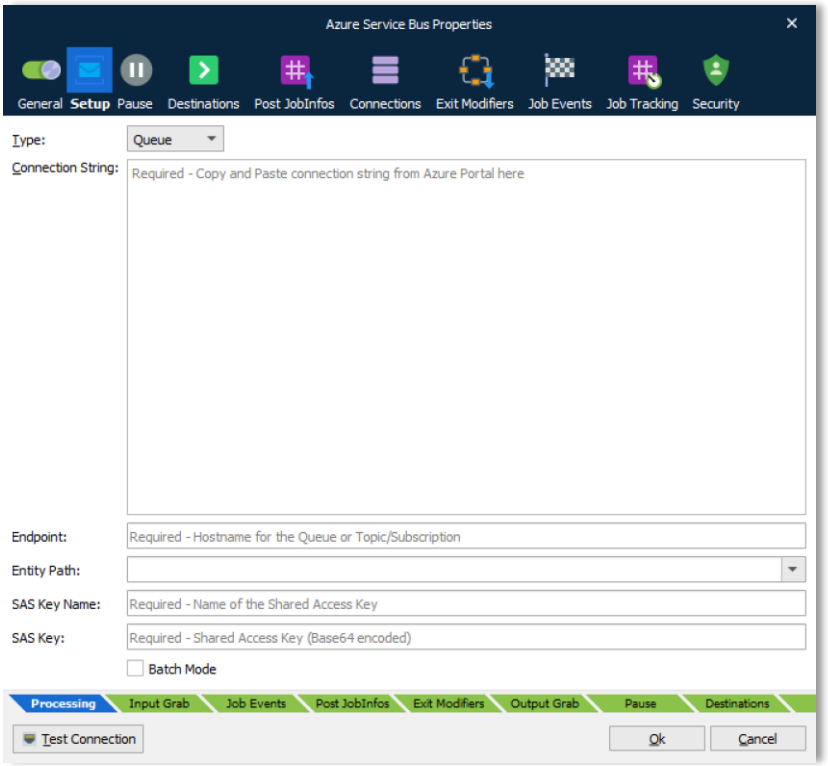
Connect HTTP based client/servers across the internet via Microsoft Azure.



Please refer to the “Lasernet 10 – Azure” manual for more information.

6.2.2 Azure Service Bus

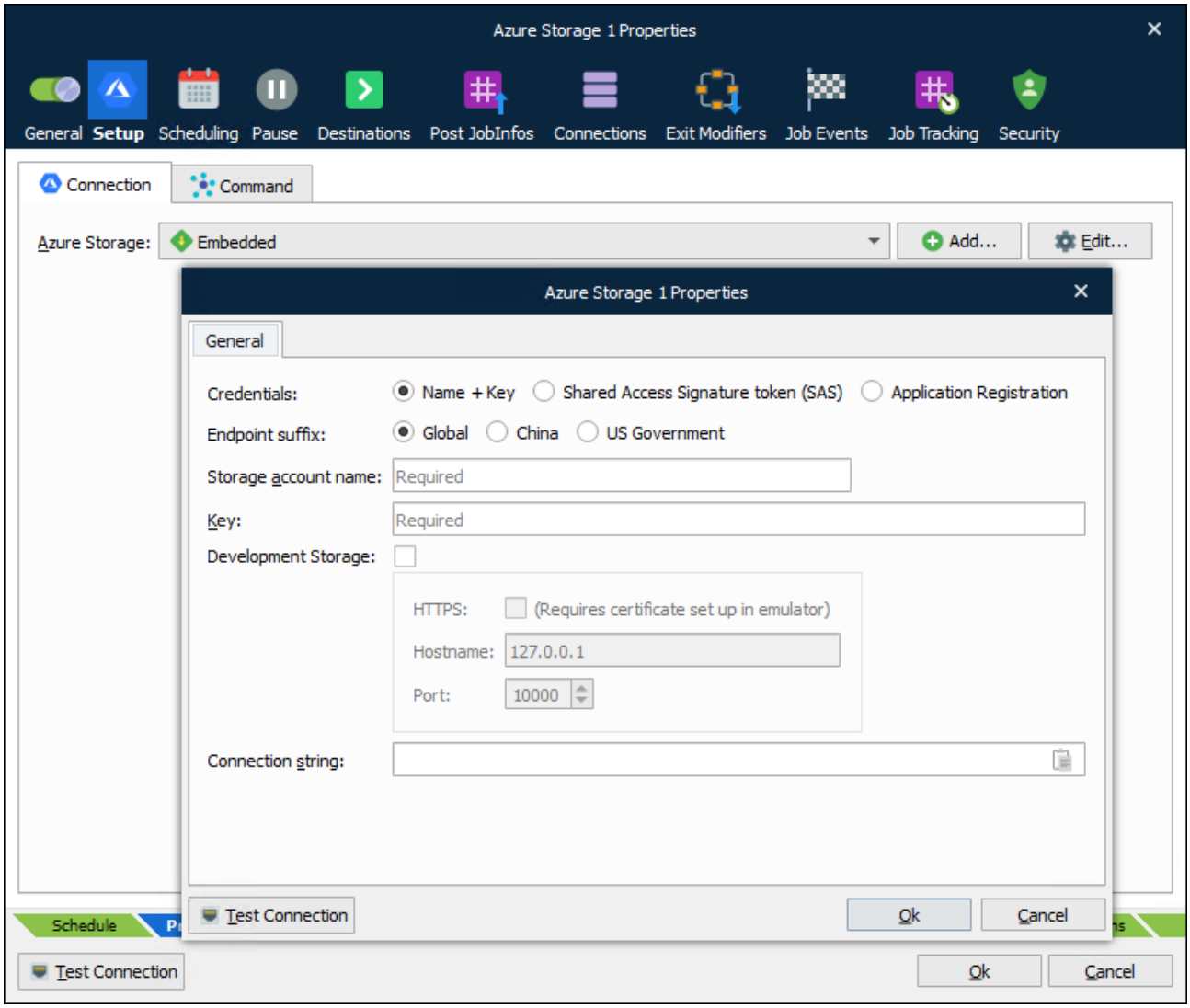
This module is used for connecting to the Azure Service Bus.



Please refer to the “Lasernet 10 – Azure” manual for more information.

6.2.3 Azure Storage

This module is used for connecting to Azure Storage.



Refer to the “Lasernet 10 – Azure” manual for more information.

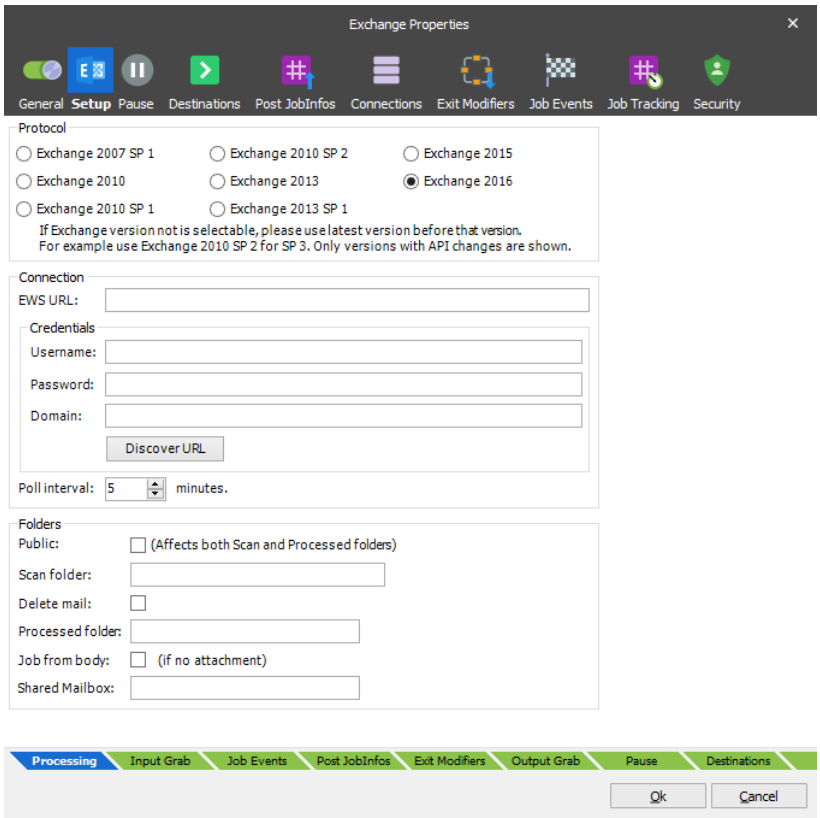
6.2.4 Dropbox

Dropbox is a service that lets you bring all your documents in to the cloud.

Important: This module is deprecated in Lasernet and will not be supported in future versions of Lasernet.

6.2.5 Exchange

Used for picking up files from a Microsoft Exchange Server. The Exchange module integrates directly with Exchange Web Services (EWS) to retrieve emails.



EWS was introduced by Microsoft in Exchange Server 2007, but will no longer receive feature updates. While the service will continue to receive security updates and certain non-security updates, product design and features will remain unchanged.

Microsoft has announced that EWS will still be available and supported for use in production environments. However, we suggest migrating to the Outlook Mail module, using Microsoft Graph API to access Exchange Online data and gain access to the latest features and functionality. The Outlook Mail module was introduced in Lasernet 9.5

Processed emails will be marked as read by Lasernet.

6.2.5.1 General

Protocol	The version number of the Exchange Server being connected to.
EWS URL	Location of the Exchange Web Services. If connecting to Microsoft Online (Cloud solution), use the following URL:

	https://red002.mail.emea.microsoftonline.com/ews/exchange.asmx or https://amsprd0310.outlook.com/EWS/Exchange.asmx If the URL is left empty, Lasernet will run an auto discover service each time the poll interval is reached. If the URL is retrieved successfully it will be logged and a connection will be established. We recommend that you add the URL stored in the log, to the EWS URL field to enable faster connection to the Exchange Server in future.
Username	Email address or name of active directory user
Password	Password for the above account
Domain	Domain name for the active directory user
Poll interval	The interval between retrieval of new emails
Public	Activate to scan in public folder instead of private folder. This will also affect the location for the processed folder if defined.
Scan folder	Folder where emails are read from. If scanning a sub folder of the Inbox, the syntax is Inbox\test
Processed folder	If specified, processed emails will be moved here. If left blank, the mail will remain in the scan folder. If processed to a sub folder to Send, the syntax is Send\test
Job from body	A Job will be created from the email body, if no attachment available.
Profile Password	If the MAPI profile requires a password, enter it here.

6.2.5.2 JobInfos

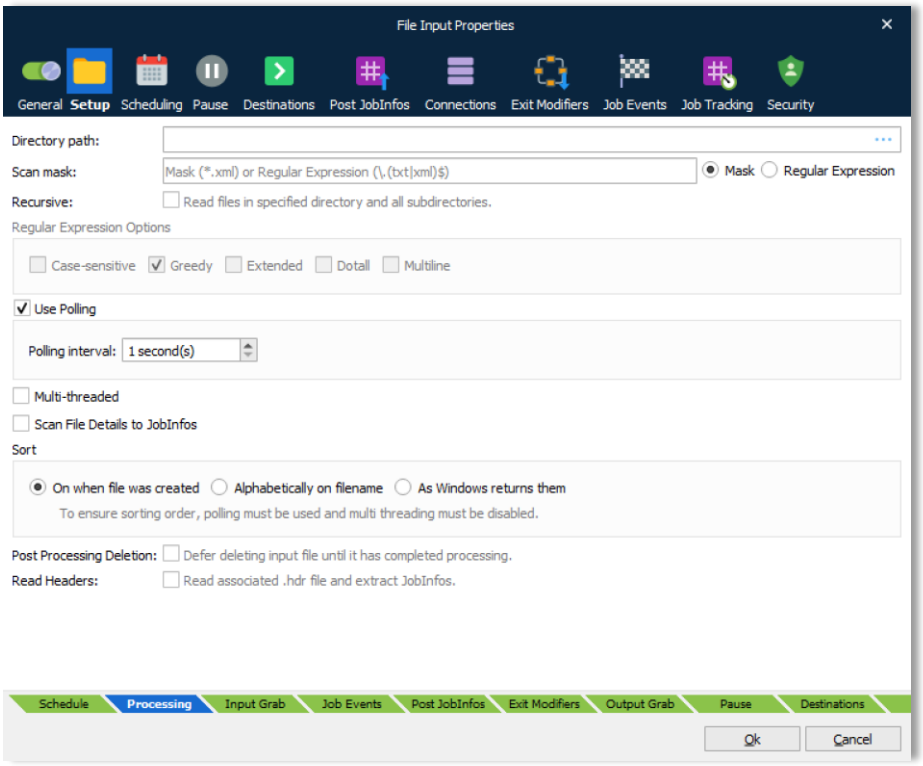
When a mail is received, a job is created for each attachment – JobData is set to the attachment and the following JobInfos are set:

InputAttachmentCount	Total number of attachments included in the incoming email.
InputAttachmentNumber	Handling attachment number for the incoming email.
InputBody	The body of the mail.
InputFilename	The filename of the attachment.
InputFromEmail	The email address of the sender.
InputFromName	The display name of the sender.

InputMessageID	Unique identifier for the incoming mail.
InputMimeType	The mimetype of the attachment.
InputReplyTo	The ‘reply to’ address of the sender for the incoming email.
InputSubject	The subject of the mail.

6.2.6 File Input

Used for picking up files from a directory for processing in Lasernet.



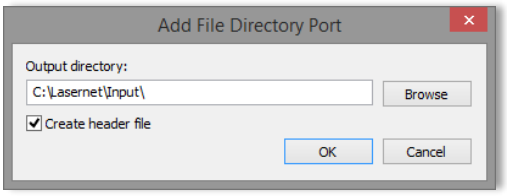
There are three main settings that must always be configured for the File Input module:

Directory path	Specifies the directory that the input module will look in to find files.
Scan mask	Filter that determines what files to pick up. Wildcard: The asterisk (*) is used for matching any characters. For example, ***.xml** will match any file with an xml extension. **Regular Expression:** If you need to match more than one type or you have a specific list of files to recognize, you can use regular expressions. An example of multiple file types: **\.(txt|xml|pdf)$**

Recursive	Activate to recursively read all files in specified directory and all subdirectories.
Regular Expression	Activate to enable **Regular Expression Options**. Offers enhanced method of setting up wildcards for matching file names.

Polling, multi-threaded and sorting are behaviours for detecting and receiving files.

Use Polling	Polling is activated by default. This means that the file input module will poll the specified directory every time the polling interval is reached. If polling is disabled, Lasernet will react to events from Microsoft Windows when a new file is available; however, this is not reliable method for network paths and makes sorting impossible, so polling is recommended.
Multi-threaded	Multi-threaded is deactivated by default. This means that the File Input module will only process one job at a time. If multi-threaded is activated, Lasernet will simultaneously process as many jobs as there are CPU cores in the server. In active mode, Lasernet cannot guarantee the output order of the processed jobs, since one job could be processed faster than another job.
Sorting	Defines the sorting order that the file input module will process the incoming jobs. To ensure a fixed sorting order, polling must be used and multi-threading must be disabled.
Scan file details to JobInfos	When this setting is enabled, Lasernet scans metadata from the file. This similar to what is shown in Microsoft Windows Explorer when right-clicking a file and selecting Properties. The names of the JobInfos will depend on the file type, but they are always prefixed by FileDetail, e.g., FileDetailAuthor. Please use the Lasernet Monitor application to view the JobInfos that are set for the file types that you are using.
Read Headers	Enable to scan for a matching .hdr file and read the embedded JobInfos. The .hdr file is used to store information about a print job created by a Windows printer and saved by the Lasernet File Directory Port.



The .hdr file is deleted after the job is read by the File Input Port.

Handling for Lasernet job format (Injob)

The internal job format in Lasernet is called **Injob** and is a zipped archive format that contains names and values for any JobInfo defined for a job. The File Input module is able to read the **Injob** format and automatically extract all JobInfo names and values stored in the archive file including JobData. After a job has been extracted by the File Input module it can be passed to other modules in Lasernet with included JobInfos.

6.2.6.1 JobInfos

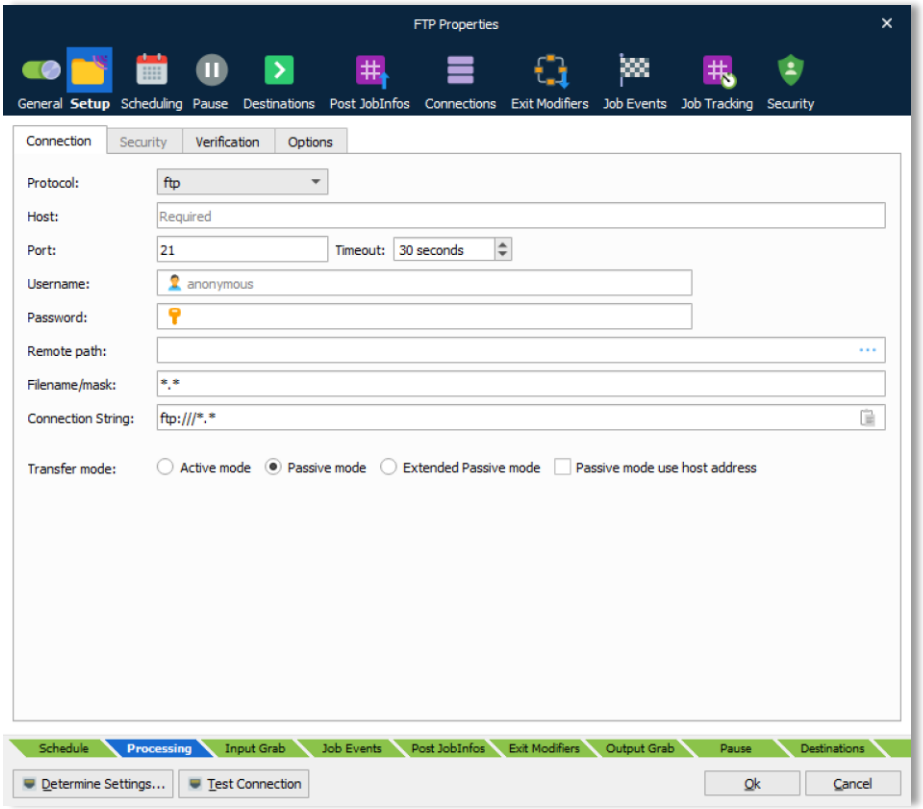
The File Input module sets some JobInfos when creating a Job.

Default	The default destination as set in the configuration.
DefaultPrinter	The default printer as set in the configuration.
Extension	The extension of the filename.
FileCreated	The time the file was created in the format yyyy-mm-dd hh:mm:ss.zzz. Not with JobInfo Scanner.
FileLastModified	The time the file was last modified in the format yyyy-mm-dd hh:mm:ss.zzz. Not with JobInfo Scanner.
FileLastRead	The time the file was last accessed in the format yyyy-mm-dd hh:mm:ss.zzz. Not with JobInfo Scanner.
Filename	The name of the file without the path.
FilenameWithoutExt	The name of the file without the extension.
FilePath	The path to the file without filename.
FileRelativePath	The relative path to the root of the file.
FileRootPath	The path to the root of the file without filename.
FullFilename	The name of the file including path.
MatchedMask	The file mask matched.
PageSeparators	The page separators as set in the configuration. Only with JobInfo Scanner.
TimeOut	The time out setting. Used only with JobInfo Scanner.

6.2.7 FTP

Used for retrieving files from FTP and FTPS servers at given time intervals. It is possible to retrieve multiple files per connection.

To connect to an FTP server, select the protocol and enter the address of the server into the host field. Add the server port number into the port field (defaults to 21 unless specified). If a username and password is required, enter it in the corresponding fields (defaults to anonymous logon if no credentials are specified). Click **Determine Settings** to validate against the specified host and retrieve information for supported ports, transfer modes, protocols, SSL types and Clear Control Channel (active/not active).



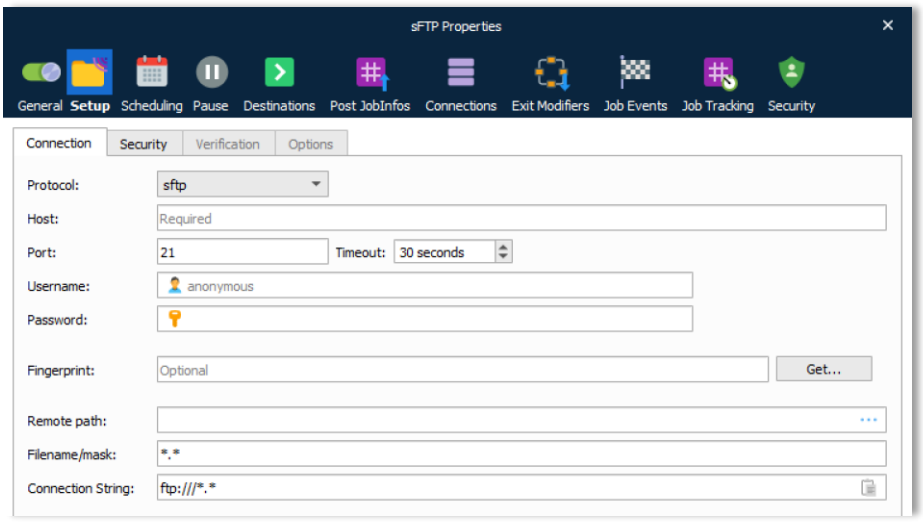
6.2.7.1 Connection

Protocol	Supported protocols are FTP, FTPS and SFTP
Host	URL for FTP. Protocol type or remote path should not be included as part of URL.
Port	Set port number used by host. Default for FTP is port 21 and for SFTP port 22.
Timeout	If the connection cannot be established before the timeout limit it will fail. Default timeout is 30 seconds.
Username and password	If user authentication is required, enter a username and password, otherwise anonymous logon will be used by default.

Remote path	The remote path on the FTP server where files need to be uploaded to (if different from the default log in directory of the FTP server). Click the Browse button to select, create or delete a folder on the remote host.
Filename/mask	Wildcards are supported for selecting one or more files to retrieve.
Transfer mode	Active mode, passive mode and Extended Passive mode are supported. In **active mode**, the FTP client opens a dynamic port, sends the FTP server the dynamic port number on which it is listening over the control stream and waits for a connection from the FTP server. When the FTP server initiates the data connection to the FTP client, it binds the source port to port 20 on the FTP server. In **passive mode**, the FTP server opens a dynamic port, sends the FTP client the server's IP address to connect to and the port on which it is listening over the control stream and waits for a connection from the FTP client. In this case, the FTP client binds the source port of the connection to a dynamic port. In **extended passive mode**, the FTP server operates exactly the same as passive mode, however it only transmits the port number (not broken into high and low bytes) and the client is to assume that it connects to the same IP address that was originally connected to.

6.2.7.2 SFTP

The SSH File Transfer Protocol (SFTP) uses a cryptographically protected connection to communicate over. Enter the port of the server into the port field if it is not port 22 (default for SFTP).

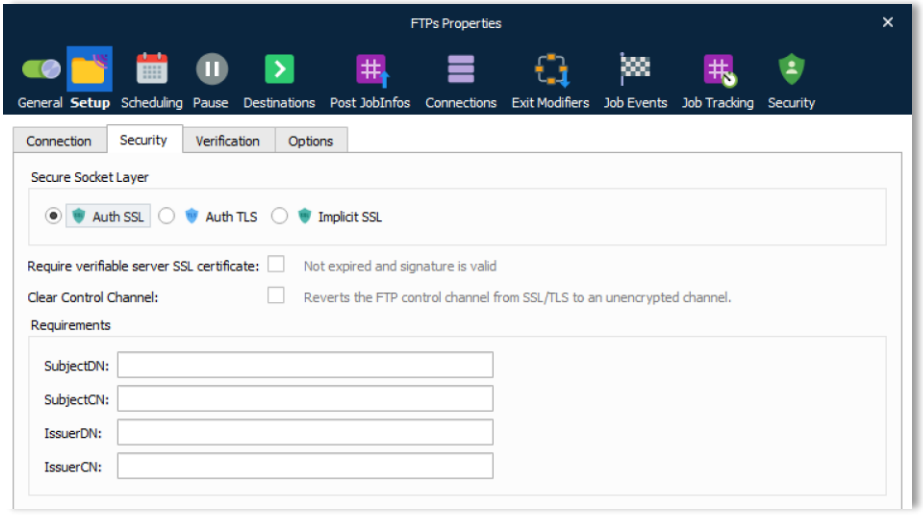


The SFTP protocol has almost identical settings to the FTP protocol, expect for the following parameters:

Transfer mode	Not available for SFTP.
Fingerprint	Before establishing a connection, the SFTP server sends an encrypted fingerprint of its public host keys to ensure that the SFTP connection will be exchanging data with the correct server. Once you have established a connection to an FTP server and are sure that it is the correct server, you should save the fingerprint information locally. This enables you to check the fingerprint information against the data you have saved every time you establish a new connection to ensure that no one is between you and the server.

6.2.7.3 FTPS

To enable the Security tab, FTPS must be selected as the protocol. Enter the port of the server into the port field if it is not port 990 (default for FTPS).

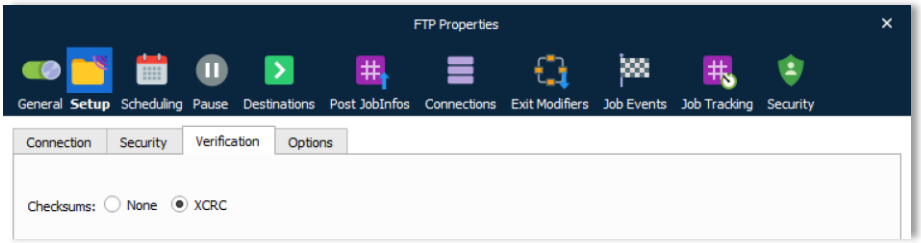


Secure Socket Layer	FTPS (SSL/TLS) is available in two incompatible modes. If using explicit FTPS, the client connects to the normal FTP port and explicitly switches into secure (**SSL/TLS**) mode with **AUTH TLS**, whereas **Implicit SSL** is an older style service that assumes SSL/TLS mode right from the start of the connection (and normally listens on TCP port 990, rather than 21). If a **verifiable server SSL certificate** is required it must be activated. Ensure that certificate is not expired and the signature is valid. Using **Clear Control Channel** enables the Firewall to open the correct ports.
Requirements	**SubjectDN:** Identifies the DN of the server's certificate **SubjectCN:** The certificate owner's common name

IssuerDN: Identifies the CA that issued the certificate

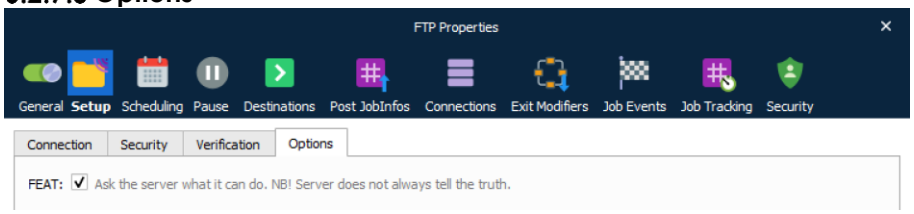
IssuerCN: The certificate issuer's common name

6.2.7.4 Verification



Checksums	A checksum file can be sent first and used for real-time error checking as subsequent files are uploaded. This is a feature or script which the FTP server must support. Lasernet does not currently handle any errors that might occur here. Alternatively, XCRC can be enabled if the server supports it. Files are automatically checked and fixed while uploading, guaranteeing correct delivery.

6.2.7.5 Options



FEAT	Active FEAT if you need to ask the server for additional commands that it supports outside the basic FTP protocol defined in RFC 959. Note: All servers don't respond accordingly to the RFC 959 standard.

6.2.7.6 JobInfos

The FTP module sets some JobInfos when creating a Job.

MatchedMask	The file mask matched.
Filepath	The URL for the file without the filename.
Filename	The name of the file without URL.
FullFilename	Name of the file including URL and filename.
Filesize	Size of the file.
Extension	Extension of the file.

FilenameWithoutExt	Name of the file without URL and without extension.

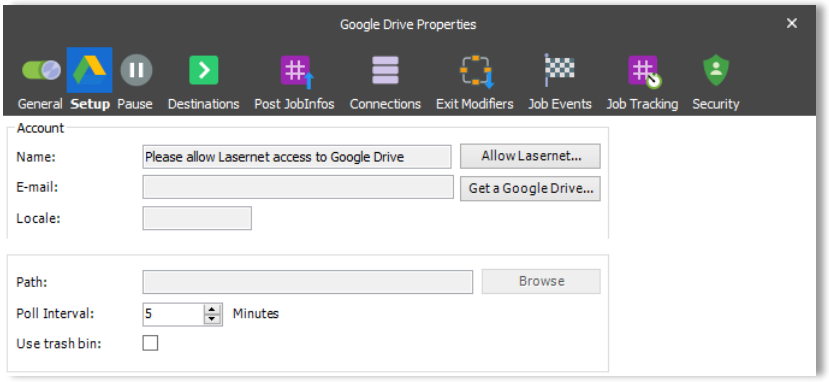
6.2.8 Google Drive

Google Drive lets you store and access your files anywhere.

Important: This module is deprecated in Lasernet and will not be supported in future versions of Lasernet.

6.2.8.1 Input

Lasernet can poll a folder on Google Drive. When a file has been downloaded it is either permanently deleted or deleted to the trash bin on Google Drive.

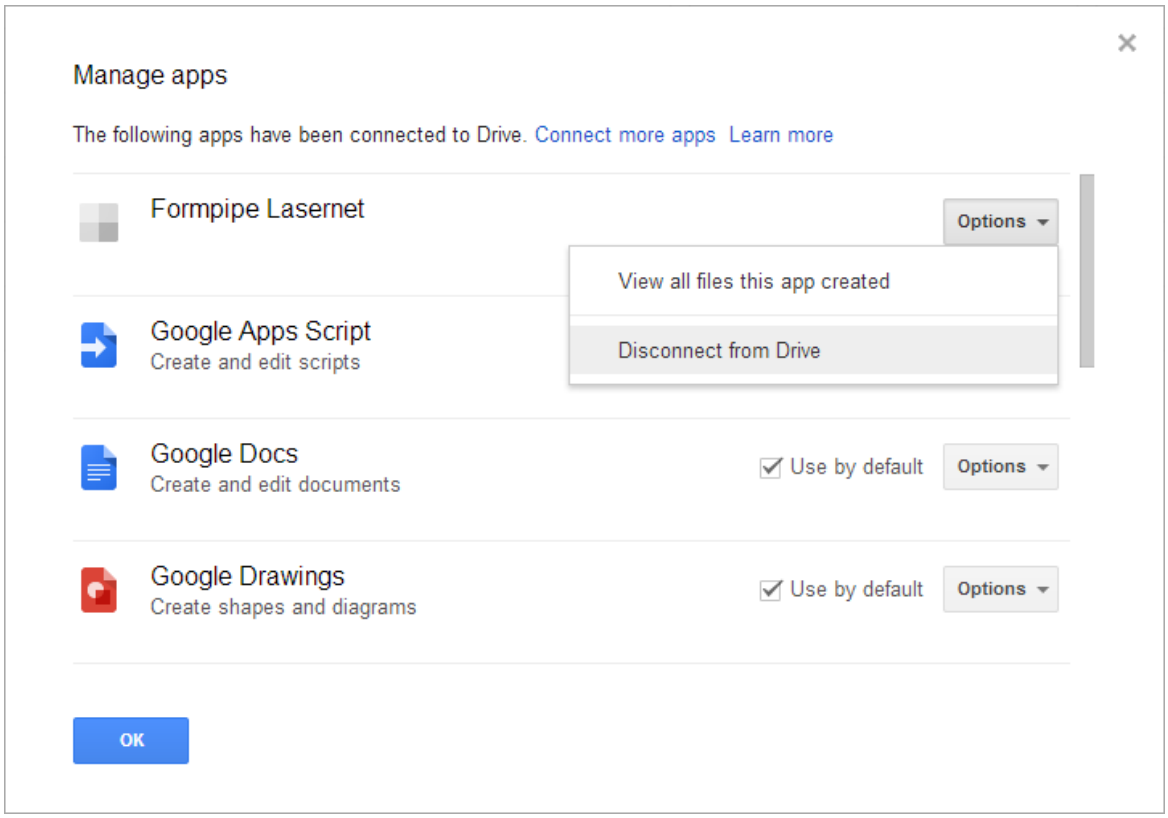


6.2.8.2 Authentication

Google Drive uses OAuth 2.0 authentication. The OAuth 2.0 authorization framework enables Lasernet to obtain limited access to Google Drive by orchestrating an approval interaction between the resource owner and Google Drive. This basically means that you need to give Lasernet access to your Google Drive.

Giving Lasernet access to Google Drive is done by pressing the “Allow Lasernet” button, which launches a web browser Window where the user can login to the Google Drive account and allow Lasernet access to it. By doing this, an encrypted token is stored in the configuration. This token is used later to regain access to Google Drive.

Revoking access is done via the web interface for Google Drive where it’s possible to disconnect an app, in this case Formpipe Lasernet, from the Drive.



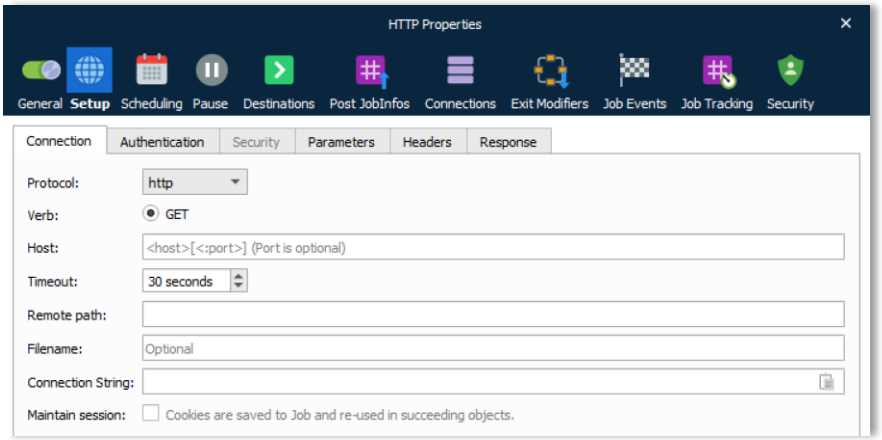
6.2.8.3 JobInfos

A Job is created per file downloaded. The following JobInfos will be set on the Job.

FileLastModified	When the file was last modified
FileCreated	When the file was created
FullFilename	Full path to the file including the filename
Filepath	Location of the file
Filename	Name of the file including extension
Extension	Extension of the file prefixed with a dot
FilenameWithoutExt	Name of the file without an extension
Filesize	Size of file in bytes

6.2.9 HTTP

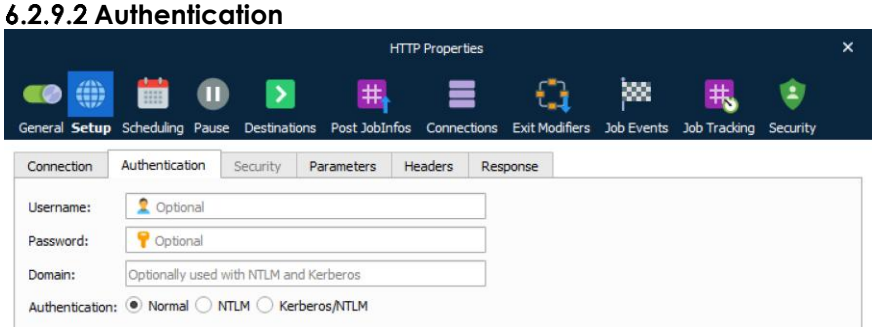
Used for retrieving a file from HTTP and HTTPS servers at given time intervals.



6.2.9.1 Connection

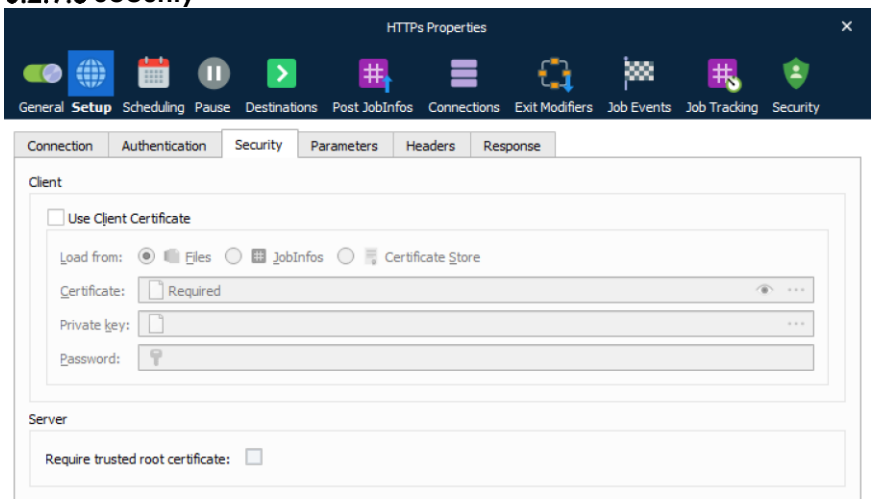
Protocol	Supported protocols are HTTP and HTTPS.
Host	URL for HTTP. Protocol type or remote path should not be included as part of the URL.
Timeout	If a connection cannot be established before the timeout limit, it will fail. Default timeout is 30 seconds.
Remote path	The remote path on the HTTP server where files need to be uploaded to (if different from the default log in directory of the HTTP server).
Filename	Wildcard support for returning one or more files to retrieve.

6.2.9.2 Authentication



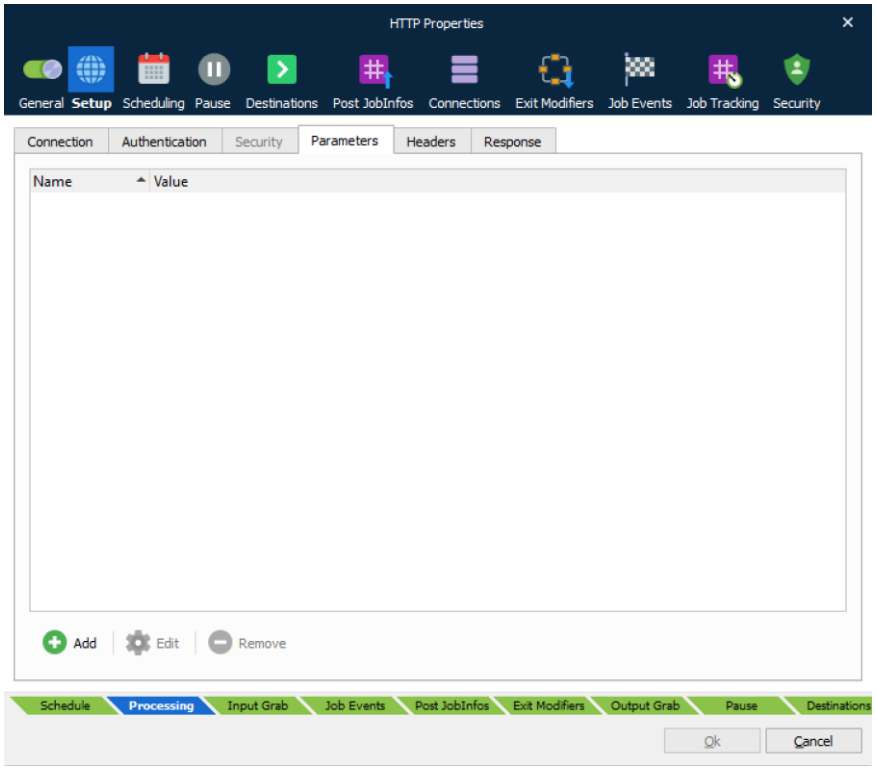
Username and password	If basic authentication is required, enter username and password.
Domain	Used with NTLM and Kerberos authentication.
Authentication	Supported types are Normal authentication, NTML and Kerberos/NTLM.

6.2.9.3 Security



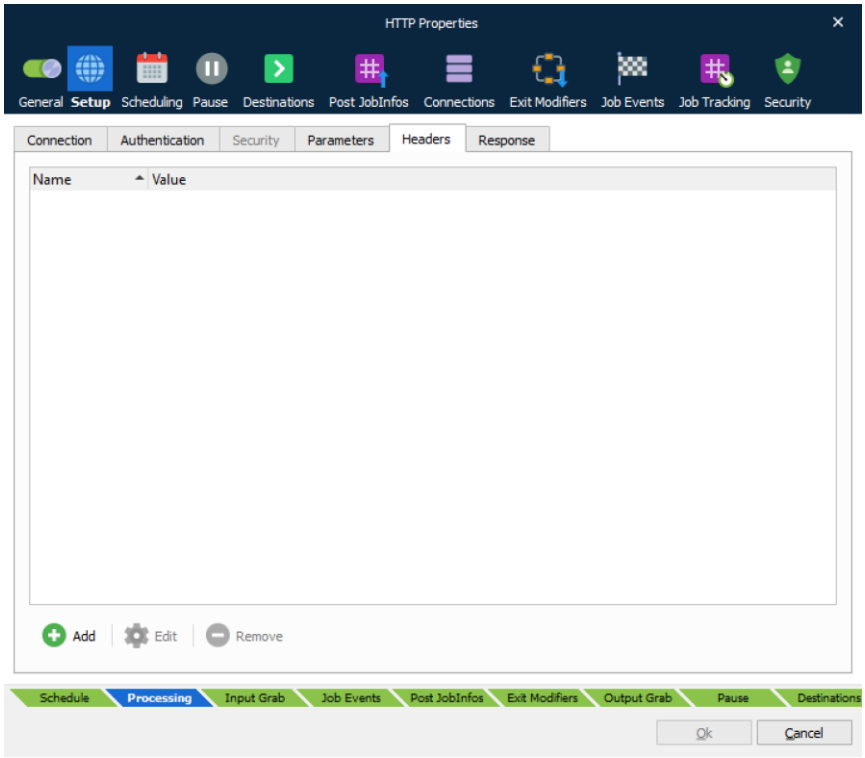
Use Client Certificate	For HTTPS it is usually not necessary to specify a client certificate and private key. If required you must activate this setting, browse for a client certificate and select a filename with one of the extensions for X.509 certificates.
Client Private Key	Browse and select a private key for the client for being authenticated.
Private Key Password	Set the password for the private key.
Subject Common Name (CN)	When enabled you must set a CN to match the subject for the host being authenticated.
Require trusted root certificate	Enabling this option will perform a range of verifications on the certificate used for HTTPS: 1. Validation of the certificate. 2. Expiration of any certificate in the chain. 3. Root certificate is trusted.

6.2.9.4 Parameters



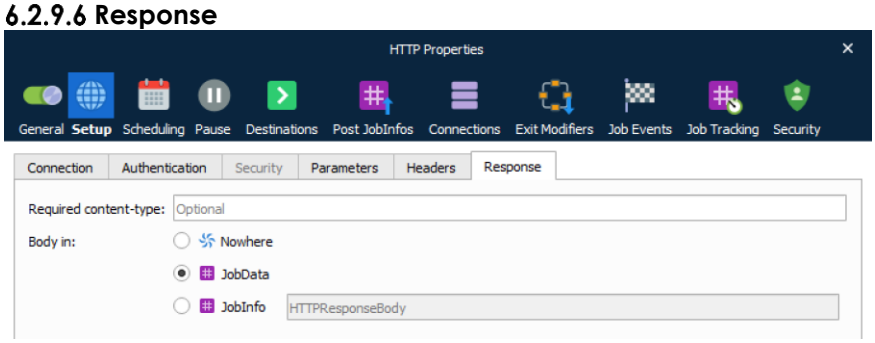
Parameters A set of parameters with values can be added to the HTTP request.

6.2.9.5 Headers



Headers	A set of headers with values can be added to the HTTP request.

6.2.9.6 Response



Required content-type	Content-type required for the document.
Body in	After receiving and interpreting a request message, the server responds with an HTTP response message. You can define where to insert the HTTP body. Select **Nowhere** to store it nowhere, **JobData** if you want to use it as job data in another module, or **JobInfo** if you want to insert the value into a specific JobInfo.

6.2.9.7 JobInfos

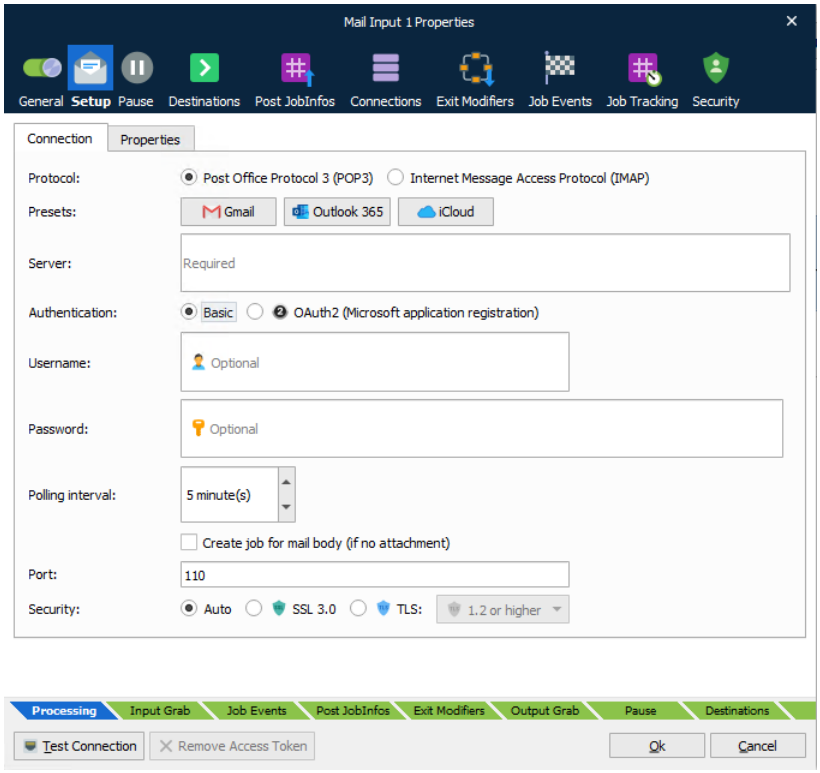
The HTTP module sets some JobInfos when creating a Job.

MatchedMask	The file mask matched.
Filepath	The URL for the file without the filename.
Filename	The name of the file without URL.
FullFilename	Name of the file including URL and extension.
Filesize	Size of the file.
Extension	Extension of the file.
FilenameWithoutExt	Name of the file without URL and without extension.
HTTPHeaderFieldName	JobInfo list containing field names for the HTTP response headers.
HTTPHeaderFieldValue	JobInfo list containing field values for the HTTP response headers.
HTTPHeader*	The headers described by **HTTPHeaderFieldName** and **HTTPHeaderFieldValue** are also stored in individual dedicated JobInfos. For example, **HTTPHeaderContent-Length**.

HTTPStatusCode	Status code for HTTP response.
HTTPStatusText	Status text for HTTP response.

6.2.10 Mail Input

Used for picking up files from an email mailbox.



The following mail servers are supported:

- Servers using the POP3 standard.
- Services using Internet Message Access Protocol (IMAP).

6.2.10.1 General

Protocol	POP3 and IMAP
Server Address	DNS name or IP address of POP3 server.
Authentication	Select **Basic** authentication or **OAuth2 (Microsoft application registration)** authentication.
Username	Username for mail account.
Password (**Basic** authentication only)	Password for mail account.
Tenant Domain (**OAuth2** authentication only)	Tenant domain of the app registration.

Client ID (**OAuth2** authentication only)	Client ID of the app registration.
Client secret (**OAuth2** authentication only)	Client secret of the app registration.
Polling Interval	Sets how often Lasernet should look for new mail.
Create job for mail body	If selected, a job will be created for the mail body.
Port	Number of SSL port, if SSL is used.
Security	Selectable choices for security protocols are **None**, **SSL 3.0** and **TLS**. Activate SSL port and set up the port number (default 465) to provide encrypted communication and secure identification. Configure the TLS protocol to use specifically version **1.0**, **1.1**, **1.2** or **1.3** of the protocol or configure it to negotiate with the server the use of a particular version or higher (for example, **1.2 or higher**).

Properties

Scan Folder	Specifies the folder to look in for incoming mail. If left blank, the Inbox folder will be used.
Processed Mail Folder	Sends mail to this folder after handling it. If left blank, the mail will remain in the incoming mail folder, but will be marked as read.
Delete mail after processing it	If checked, will delete the mail after handled by Lasernet.

6.2.10.2 OAuth2 Configuration

To set up OAuth2 authentication, you must complete tasks in Microsoft Entra ID and Exchange in addition to configuring this Lasernet module.

Follow this overall process:

1. In Entra ID, create an application registration for Lasernet and then grant it the following permissions:
   - For POP access: `POP.AccessAsApp`
   - For IMAP access: `IMAP.AccessAsApp`
   - For SMTP access: `SMTP.SendAsApp`
2. In Entra ID, get tenant admin consent for the Lasernet application registration to access Exchange mailboxes through POP, IMAP, or SMTP.
3. In Exchange, use the `New-ServicePrincipal` cmdlet to register the service principal.
   - **Note:** This step must be completed by an Exchange administrator.
4. In Exchange, use the `Add-MailboxPermission` cmdlet to grant the application the necessary access to specific mailboxes.
   - For example: `Add-MailboxPermission -Identity "john.smith@example.com" -User 3f2504e0-4f89-11d3-9a0c-0305e82c3301 -AccessRights FullAccess`

Note: The process above summarises the following Microsoft documentation. For full instructions and accompanying examples, go to https://learn.microsoft.com/en-us/exchange/client-developer/legacy-protocols/how-to-authenticate-an-imap-pop-smtp-application-by-using-oauth#use-client-credentials-grant-flow-to-authenticate-smtp-imap-and-pop-connections

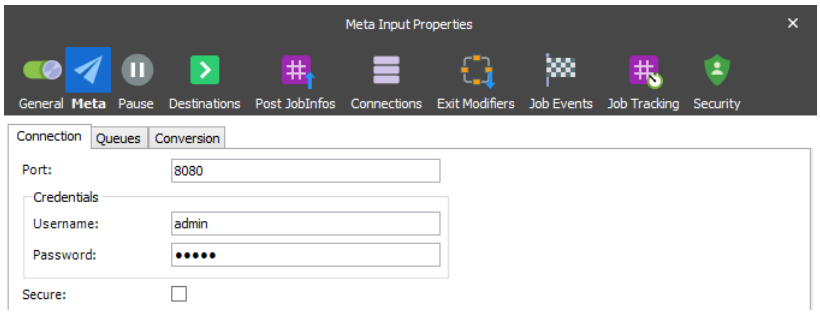
6.2.10.3 JobInfos

When an email is received, a job is created for each single attachment – JobData is set to the contents of the attachment and the following JobInfos are set if available:

Extension	Extension of the file prefixed with a dot.
InputAttachmentCount	Total number of attachments included in the incoming email.
InputAttachmentNumber	Handling attachment number for the incoming email.
InputBody	The plain text content of the mail body if any.
InputBodyHTML	The HTML contents of the mail body if any.
InputFromName	The display name of the sender.
InputFromEmail	The email address of the sender.
InputHeaderFieldName and **InputHeaderFieldValue**	A pair of JobInfo lists that contain the names and values of the custom mail headers received by this module. Each header's name is an item in InputHeaderFieldName. That header's value is the corresponding item in InputHeaderFieldValue. For example, if InputHeaderFieldName[5] = X-Priority, the value of the X-Priority mail header (in this example, 3) is in InputHeaderFieldValue[5].
InputMimeType	The mimetype of the attachment.
InputFilename	The filename of the attachment.
InputLongFilename	The long filename of the attachment.
InputMessageID	Unique identifier for the incoming mail.
InputReplyTo	The ”reply to” address of the sender for the incoming email.
InputSubject	The subject of the mail.
InputToEmail	The email address of the receiver.
InputToName	The display name of the receiver.

6.2.11 Meta Input

To allow Lasernet Meta to connect to the Lasernet Server, a Meta Input Queue must be running. Configure the listening port, authentication and queues type in the Meta Input properties.



Please refer to the “Lasernet 10 – Meta” manual for more information.

JobInfos

When a document is received a job is created. The following JobInfos are set, if available:

Autofill	Available when the Autofill button  is activated from the Lasernet Meta application. Value is set to 1.
ClientUserName	Username for the login client as defined in the Lasernet Meta → Tools → Client settings. When running with default credentials the value will be left empty.
DocumentName	The name of the print job as set by the Windows Print Spooler, when the document is printed to Lasernet Meta. Or, the name of the file as set by the Windows File System, when the document is retrieved via polling in Lasernet Meta.
DriverName	The name of the printer driver as set by the Windows Print Spooler, when the document is printed to Lasernet Meta.
Extension	The extension is always set to .pdf if the document is printed to Lasernet Meta or the extension of the filename if the document is retrieved via polling in Lasernet Meta.
FQDN	Fully qualified domain name for the user running Lasernet Meta.
MachineName	The machine name for the user who has sent the document.
MetaQueue	The name of the meta queue from which the document has been sent.
PreviewJobData	Contains a copy of the document data in Autofill mode.
PreviewExtension	Contains the extension of the document in Autofill mode.

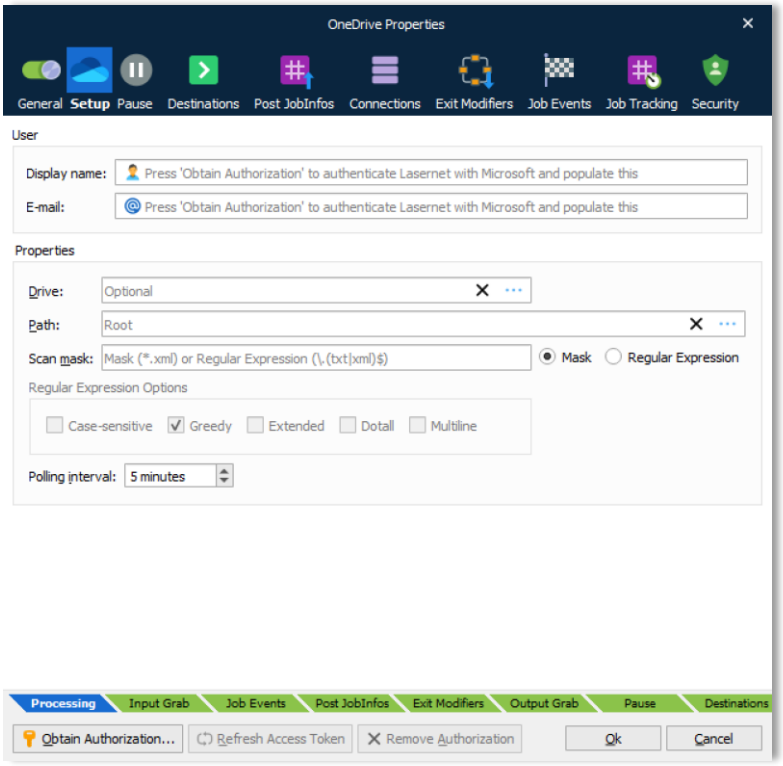
UserName	Name of the user running Lasernet Meta.
UserDomainName	Domain name for the user running Lasernet Meta.

6.2.12 OneDrive

Easily store and share documents in the cloud. Microsoft OneDrive works with Office, so it's easy to create, edit, and share your documents.

6.2.12.1 Input

Lasernet can poll a folder, for a Microsoft OneDrive, in a given polling interval defined in minutes. When a file has been downloaded it is hard deleted. A Job is created per file downloaded.



6.2.12.2 How to access Microsoft OneDrive

13. Click **Obtain Authorization** in the OneDrive dialog box to create a new connection.
14. You get redirected to the Microsoft website to **authorize OneDrive for Lasernet**.
15. **Enter** your Microsoft credentials.
16. **Authorize** the access request.
17. The authentication window closes automatically and your **Display name** and **E-mail** are added to the dialog.
18. In the Properties you can set your **Drive** and **Path** for where to poll for incoming files.

Doing this stores an encrypted token in the configuration. This token is then used to regain access to the OneDrive later.

6.2.12.3 JobInfos

The following JobInfos will be set for the incoming Job.

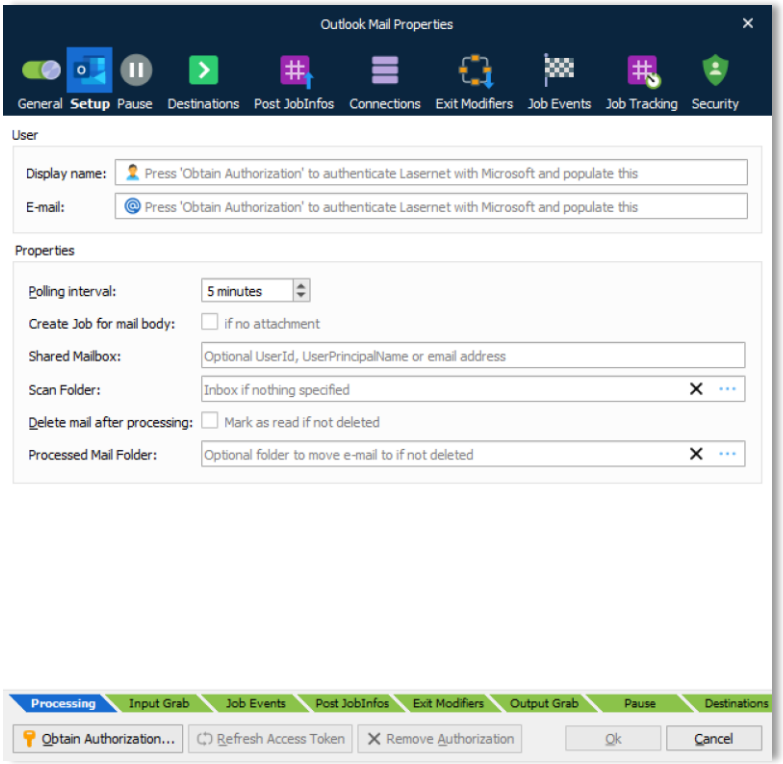
FileLastModified	When the file was last modified
FileCreated	When the file was created
FullFilename	Full path to the file including the filename
Filepath	Location of the file
Filename	Name of the file including extension
Extension	Extension of the file prefixed with a dot
FilenameWithoutExt	Name of the file without an extension
Filesize	Size of file in bytes

6.2.13 Outlook Mail

The Outlook Mail module integrates directly with Office 365 to retrieve emails. It is an alternative to the Exchange module using Exchange Web Services (EWS) introduced by Microsoft in Exchange Server 2007. EWS will longer receive feature updates, while the service will continue to receive security updates and certain non-security updates.

Microsoft has announced that EWS will still be available and supported for use in production environments, but is deprecated in Lasernet and will no longer be supported in future versions.

We recommend the Outlook Mail module, using Microsoft Graph API, to access Exchange Online data.



6.2.13.1 Obtain authentication

Click **Obtain Authorization** to access the account for your organization and to autofill the Display name and E-mail fields.

The Lasernet Outlook Mail module will obtain authorization to:

- Maintain access to data to which you have granted access
- Configure individual permission to send mail on behalf of yourself
- Configure API permission to send mail on behalf of others
- Sign you in and read your profile

6.2.13.2 General

Display name Display name for the account (click **Obtain Authorization** to autofill this field).

E-mail	E-mail address for the account (click **Obtain Authorization** to autofill this field).
Polling interval	The interval between retrieval of new emails
Create job for mail body	A Job will be created from the email body, if no attachment available.
Scan folder	Folder where emails are read from. If scanning a sub folder of the Inbox, the syntax is Inbox\test
Delete mail after processing it	Activate to delete mail after processing otherwise the email will be marked as read.
Processed Mail Folder	If specified, processed emails will be moved to this folder. If left blank, the mail will remain in the scan folder. If processed to a sub folder to Send, the syntax is Send\test

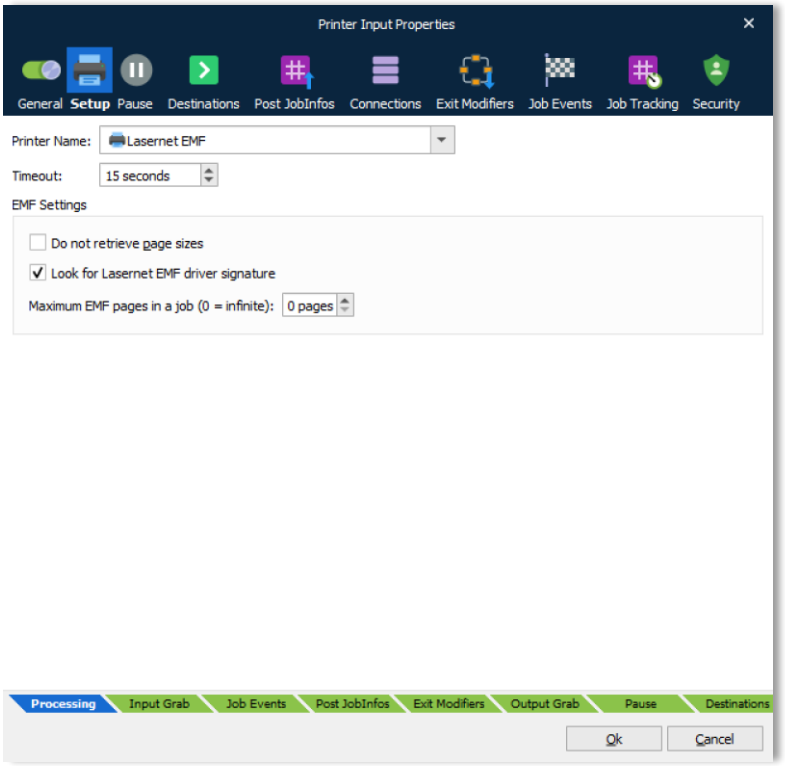
6.2.13.3 JobInfos

When a mail is received, a job is created for each attachment – JobData is set to the attachment. The following JobInfos are set:

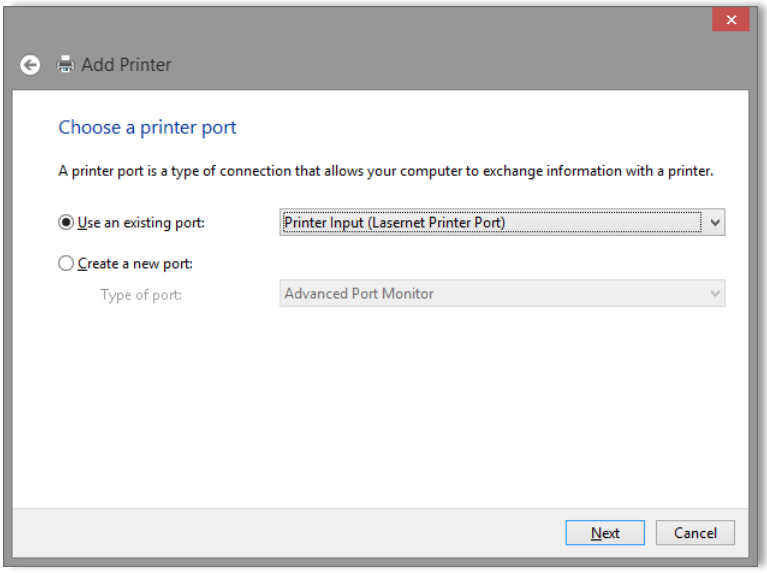
InputAttachmentCount	Total number of attachments included in the incoming email.
InputAttachmentNumber	Handling attachment number for the incoming email.
InputBody	The body of the mail.
InputFilename	The filename of the attachment.
InputFromEmail	The email address of the sender.
InputFromName	The display name of the sender.
InputHeaderFieldName and **InputHeaderFieldValue**	A pair of JobInfo lists that contain the names and values of the custom mail headers received by this module. Each header's name is an item in InputHeaderFieldName. That header's value is the corresponding item in InputHeaderFieldValue. For example, if InputHeaderFieldName[5] = X-MS-Has-Attach, the value of the X-MS-Has-Attach mail header (in this example, Yes) is in InputHeaderFieldValue[5].
InputMessageID	Unique identifier for the incoming mail.
InputMimeType	The mimetype of the attachment.
InputReplyTo	The 'reply to' address of the sender for the incoming email.
InputSubject	The subject of the mail.

6.2.14 Printer Input

Used for receiving print from the Microsoft Windows spooler system.



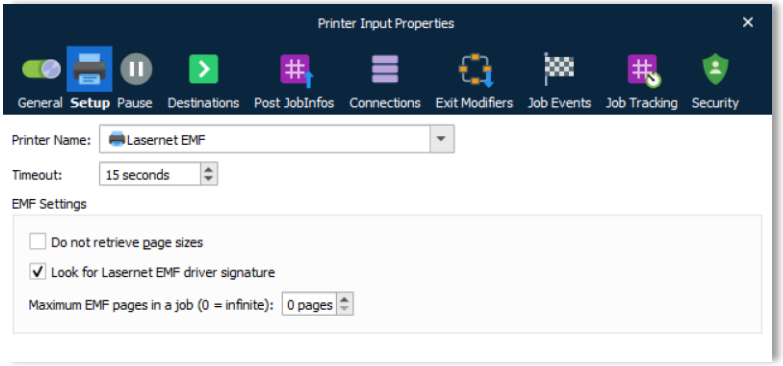
The logical name of the Print Input module is displayed on the printer port list for a Windows printer:



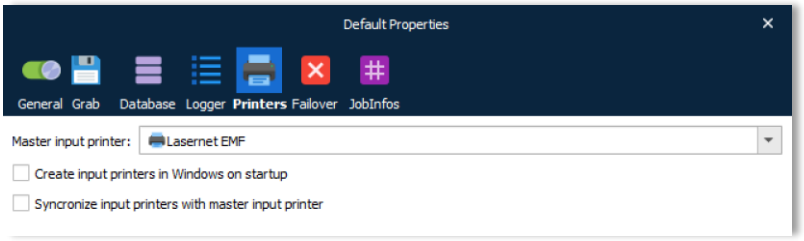
Changing the Printer Name will not immediately update the port list in Windows. The list will be updated next time the configuration is transferred to the Lasernet Service. Lasernet Printer Ports that are not in use will automatically be deleted from the printer port list.

6.2.14.1 Printer Setup

In the Printer Setup tab, you can select a Windows printer name from the drop-down list. Select the same Windows printer as you have connected the Lasernet Printer Port to.



Selecting a Printer Name is only essential if you have defined a Master input printer in the server settings and want to automatically create or synchronize input printers in Windows, at Lasernet service startup.



Read more about the master input printer in the chapter about Servers → Master Printer functionality.

The Printer Name in the Printer Input module is only queried when using the master printer functionality. We recommend selecting the same Windows printer as you have connected the Lasernet Printer Port to.

The Timeout will close the connection between the Windows printer port and Lasernet Print Input module if a closing print job message has not been received from Windows in the defined number of seconds. This will prevent the print queue from hanging in case of a printing error.

EMF Settings

The EMF Settings are only used if a Windows printer has been installed with a Lasernet EMF driver and the job was not processed by the Lasernet Print Processor.

“Do not retrieve page sizes” will prevent the Printer Input module from creating a list of system defined JobInfos named NumInputPages, InputPageWidth and InputPageHeight.

“Look for Lasernet EMF driver signature” will look for a signature in Lasernet EMF Driver instead and retrieve a list of additional JobInfos if available in the print job.

In earlier versions of Lasernet, not detecting the pages sizes or looking for a driver signature in a print job, could improve speed, but its effects are negligible in Lasernet 10.

“Maximum EMF pages in a job” can be used to split up a spool job into chunks. The default setting is 0 (zero) and means that the incoming spool job will be processed as one large job. If the size of EMF spool job is

larger than the available memory in the server, use this setting to break the job up into smaller, more manageable chunks.

6.2.14.2 JobInfos

DefaultPrinter	The default printer entered in the configuration.
Default	The default destination entered in the configuration.
WinPrinterName	The name of the printer that the job was printed on.
DocName	The name of the print job as set by the Windows Print Spooler. If empty, the default value set by Printer Input module is *Unnamed – Lasernet document*.
DataFormat	The format of the data that is generated by the printer. If the DataFormat is empty, the Printer Input module will set it to EMF by default.
DataPrinterName	The name of the Windows print queue.
MachineName	The name of the machine on which the job was printed.
UserName	The name of the user who printed the job.
NotifyName	The name of the user who should be notified of print progress.
Priority	The priority of the print job.
DataTotalPages	The total number of pages in the print job.
PrintProcessor	The name of the print processor that handled the job.
DriverName	The name of the driver that handled the print job.
NumInputPages	Total number of physical pages in the EMF spool job as received by the Lasernet EMF driver.
InputDefaultSource	The value of the selected default source for the incoming document. If available, the value is presented as a number.
InputDuplexMode	The value of the selected duplex mode for the incoming document. Known values are Simplex, Horizontal and Vertical.
InputPageWidth	Page width for each physical page in the EMF spool jobs as received by the Lasernet EMF driver. Information will be written in to an array if the job has multiple pages.
InputPageHeight	Page height for each physical page in the EMF spool jobs as received by the Lasernet EMF driver. Information will be written in to an array if the job has multiple pages.

6.2.15 Scheduler

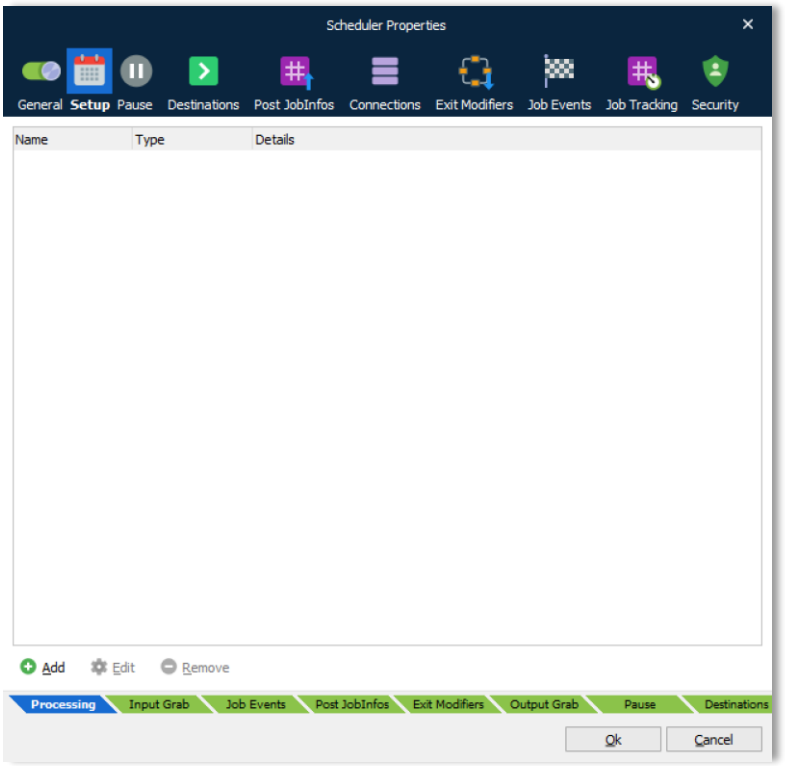
The Scheduler Input module is used for initiating actions in Lasernet at predefined times. It is primarily used for checking whether data is available from a database. This is done by creating a database command and attaching it as an exit modifier to the Scheduler Input module.

On the **Setup** tab of the Scheduler Input's **Properties** window, you specify the times when this module will run. On that tab, click **Add** to configure a scheduler event.

You specify the schedule for this module in the same way that you control the schedule by which other modules empty job queues. Refer to the scheduler event information in section 12.2 Scheduling Jobs.

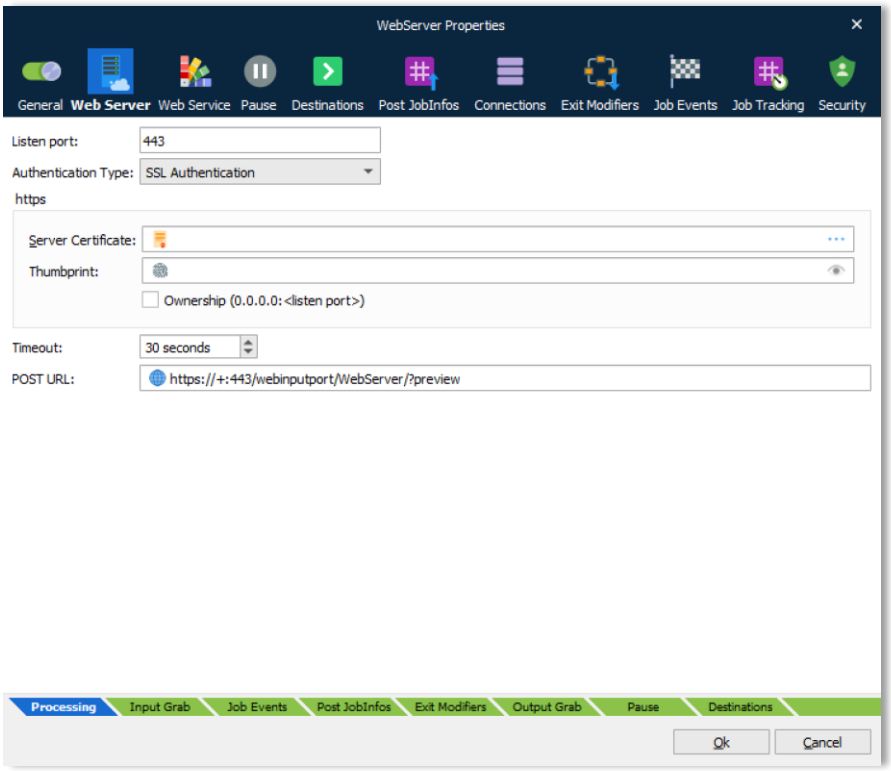
6.2.15.1 JobInfos

Each time the scheduler reaches a run point, it creates an empty job and sets the SchedulerQueue JobInfo to the name of the queue that generated the event – e.g. *Every Now and Then*.



6.2.16 Web Server

This Input module works as a web server / web service that can accept data for processing in Lasernet. It can also accept a request for data preview.



6.2.16.1 Web Server

The Web Server module acts as an embedded web server, listening on the port specified in the Web Server settings.

To deliver data to a Web Server module, an HTTP POST request must be sent to the correct URL. For a machine named “master”, that would be http://master:80/webinputport/PORTNAME, where PORTNAME is the name of the Web Server module. The actual document is delivered as the HTTP POST data. The content-type of the post data is currently ignored, but we recommend that it is set to application/x-Lasernet.

HTTPS uses port 443 by default; https://master:443/webinputport/PORTNAME

The HTTP response status will be “200 OK” if delivered successfully, “404 Object Not Found” if the port does not exist, “503 Service Unavailable” if Lasernet is currently booting or shutting down, and “500 Internal Server Error” if there is a problem with Lasernet.

When the status is 200, the response message is a text/plain document holding the JobID for the job inside Lasernet.

Lasernet can be used for sending back a document to the client, which can then be previewed in the browser e.g., a print result. This is handled by using a query parameter named preview, as in this example:

http://master:80/webintputport/PORTNAME?preview.

Lasernet will then set the JobInfo PreviewMode to 1, which can be used for deciding which destination to use in the Form Engine.

When the built-in destination “Preview” is used, it instructs Lasernet to send the document back to the client. Lasernet finds the open connection and uses it for sending the document back to the client using the response status “200 OK”.

By setting the JobInfos “HTTPStatusCode” and “HTTPStatusText” you can overrule the status code and status text set by system. By default, the content type of the preview result is “application/pdf”, but can also be overruled by setting the JobInfo “MimeType”.

It is possible to specify additional JobInfo values by adding them to the URL query. For example, to set JobInfo User to Joe and JobInfo Machine to My Machine, use the following URL: http://master:80/webinputport/PORTNAME?User=Joe&Machine=My+Machine.

The selected certificate will always be added if it does not exist prior to the start-up of the Web Server.

Ownership means that the certificate will be removed again when the Web Server stops, otherwise it will remain registered.

Default Destination is referred to as *Default* when editing destinations in other modules.

The Web Server module adds two JobInfo lists (RequestHeaderNames and RequestHeaderValues) in which it stores the headers received in the request. It also adds a JobInfo for each request parameter.

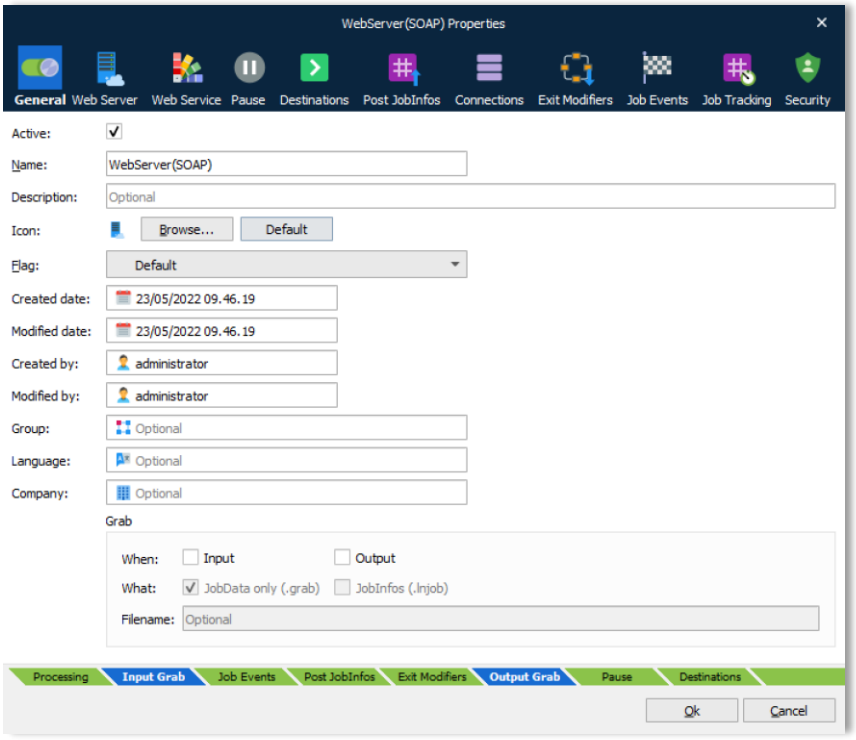
For example, if the request has headers `Content-Length: 11` and `Content-Type: application/text` and a parameter called `City` whose value is `Cambridge`, Lasernet creates the following JobInfos:

JobInfo	Value
RequestHeaderNames[0]	Content-Length
RequestHeaderValues[0]	11
RequestHeaderNames[1]	Content-Type
RequestHeaderValues[1]	application/text
City	Cambridge

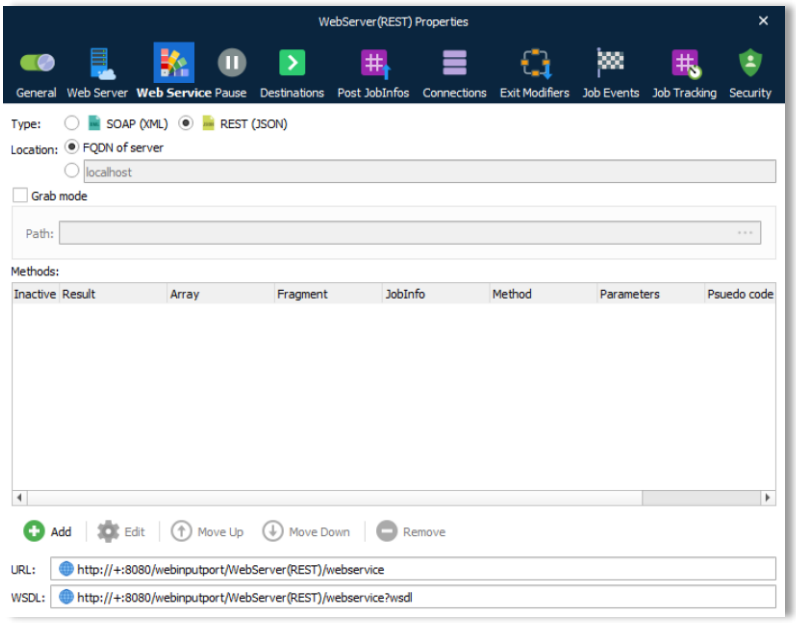
6.2.16.2 Web Service (act as server)

See Web Service chapter for more info about setting up web services on the server side.

SOAP

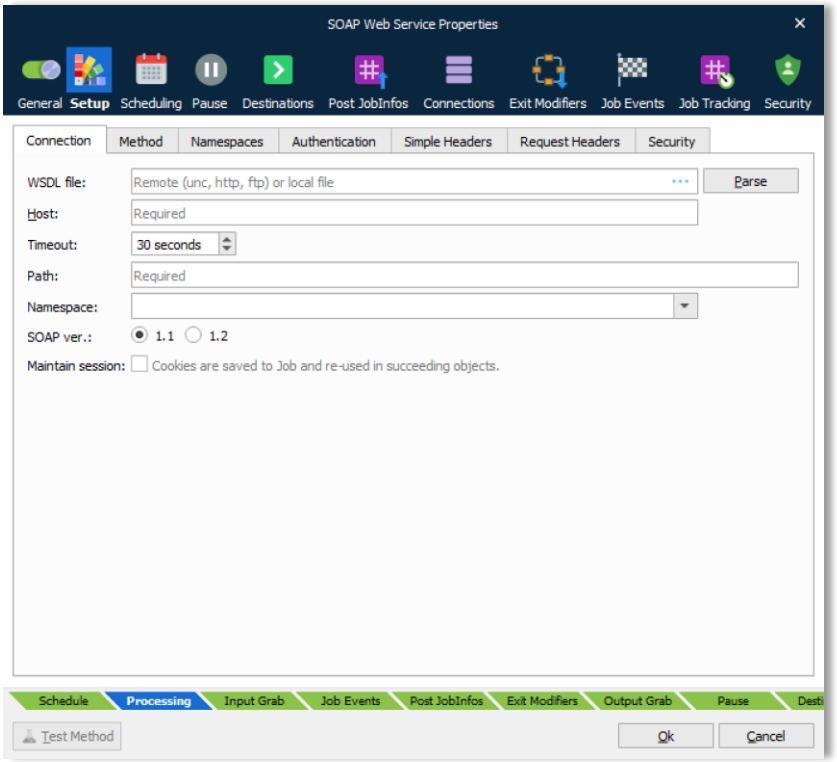


REST



6.2.17 SOAP Web Service (act as client)

This module is able to call methods published in a Web Service (act as a client).

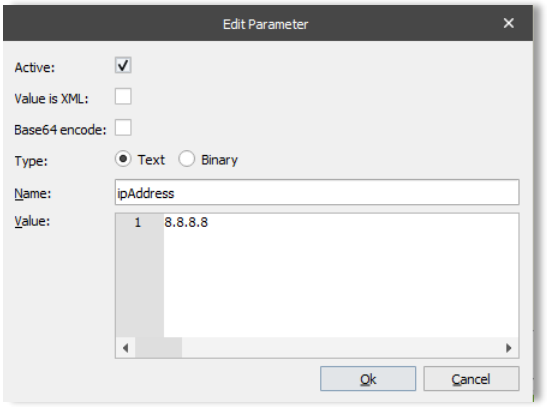


Connection and Method	Information about where the Web Service resides and the method in which it is called. Web Services Description Language file describes what the service can do. Lasernet has support for parsing WSDL files from both the hard drive and web server, as shown in the example above. When Lasernet parses the WSDL file it gathers information about namespace, methods available and their parameters. Not all WSDL files are currently supported and the parsing of one is not a requirement to build a request. 'Return in' tells Lasernet where to put the result of the Web Service request – either in JobData or in a JobInfo. All parameters are essentially strings inserted into an XML request. The Type can assist you in understanding the format of the string. 'Value is XML' tells Lasernet to insert the string as an XML fragment rather than a string, so escaping is done properly.
Namespaces	Gives support for specifying additional namespaces.
Authentication	Enables support for supplying credentials.

Simple Headers This table makes it possible to supply key value pairs in the header of the request. Some Web Services require this for simple security/authentication.

Security When using HTTPS, an optional client certificate can be configured. It is usually not necessary to specify a client certificate and private key. If required you must activate the “Use Client Certificate”, browse for a client certificate and select a filename with one of the extensions for X.509 certificates.

Parameters

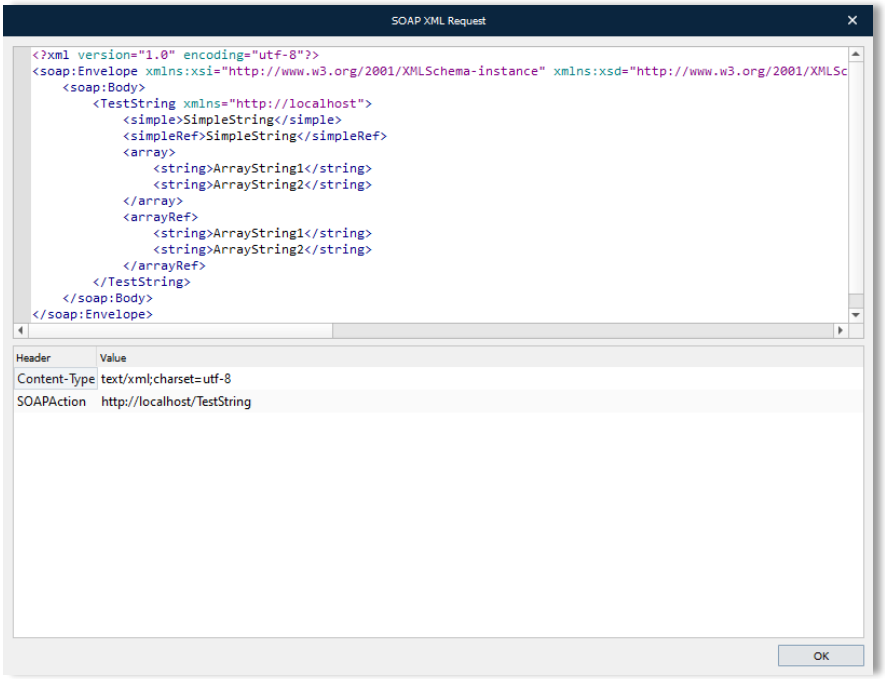


Simple parameters can be setup in the Parameters dialog. Complex parameters (data structures) must use XML as the value and ‘Value is XML’ must be checked.

Both Text and Binary values (coming from JobInfo) can be automatically Base64 encoded (which is how binary data is carried in the SOAP envelope).

JobInfo substitution is possible using `#JobInfoName#` format.

View Request

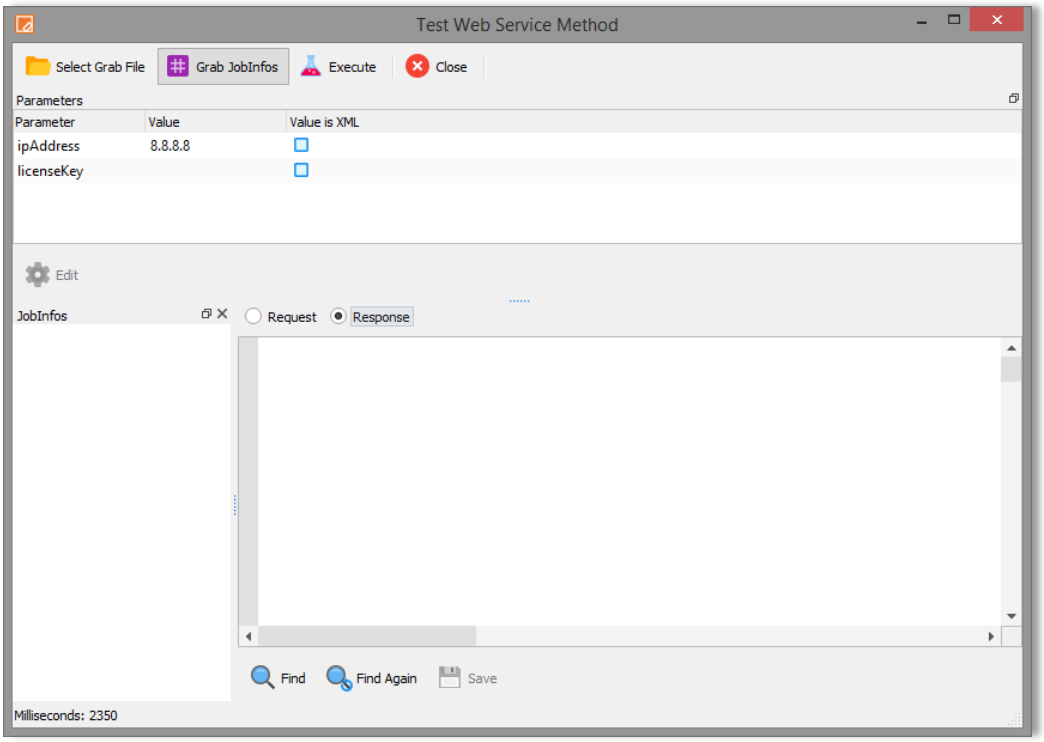


View Request shows the request and headers being sent to the web server hosting the service.

Actual JobInfo substituted parameters can be viewed in the Test Method dialog on the Request page after Executing method.

Test Method

From the Developer it is possible to test if the configured request is correct.



The result either shows the returned XML from the Web Service or information about any error that might have occurred.

The XML can be used in Form Engine or parsed as needed.

When using JobInfos as values for parameters, it is possible to load a grab .lnjob file and use or change JobInfos in it to test how the web service responds.

SOAP Faults

In case of an HTTP 500 error (internal server error) the following 3 JobInfos are set; WebServiceFaultCode, WebServiceFaultString and WebServiceDetail.

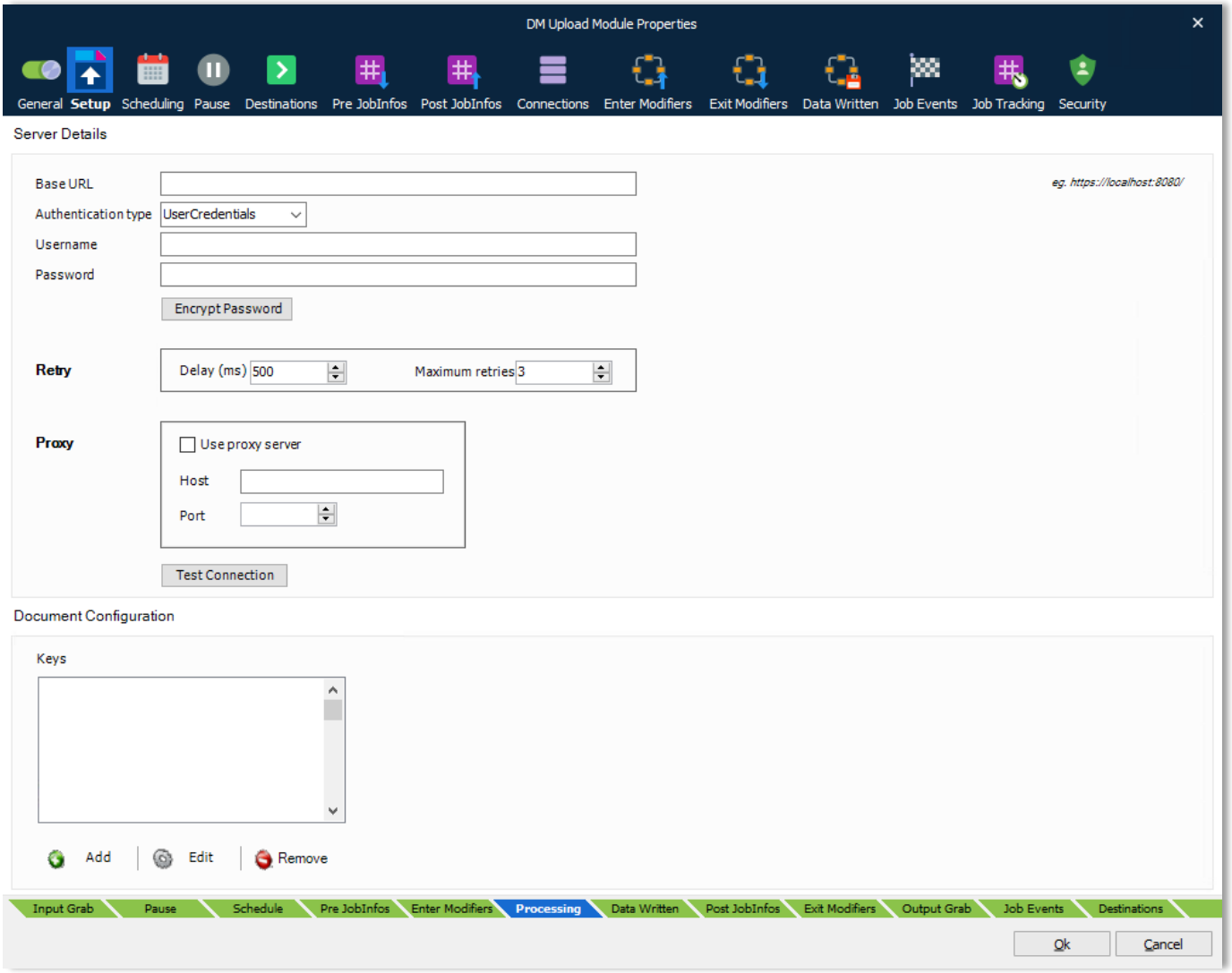
6.2.17.1 JobInfos

WebServiceFaultCode	The faultcode element is intended for use by the software to provide an algorithmic mechanism for identifying the fault.
WebServiceFaultString	The faultstring element is intended to provide a human readable explanation of the fault and is not intended for algorithmic processing.
WebServiceDetail	The detail element is intended for carrying application specific error information related to the Body element. The absence of the detail element in the Fault element indicates that the fault is not related to processing of the Body element.

6.3 Engines

Many Lasernet features are available as both modifiers and engines. The best practice is to use modifiers, unless jobs are split/merged (combined).

6.3.1 Autoform DM Upload (as engine)



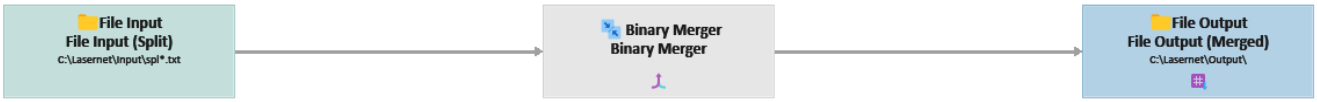
The module is designed to provide a simple, yet effective method for archiving documents and associated metadata to Autoform DM. For more information, please read the **Autoform DM Upload Module** section in the **Output Modules** chapter.

6.3.2 Binary Merger

This engine combines multiple binary jobs into a single job.

The JobInfo named ‘JobData’ will contain the data for all the combined jobs. It will be attached to the JobInfos that were created in the first job. JobInfos for all other Jobs expect the first job will be deleted.

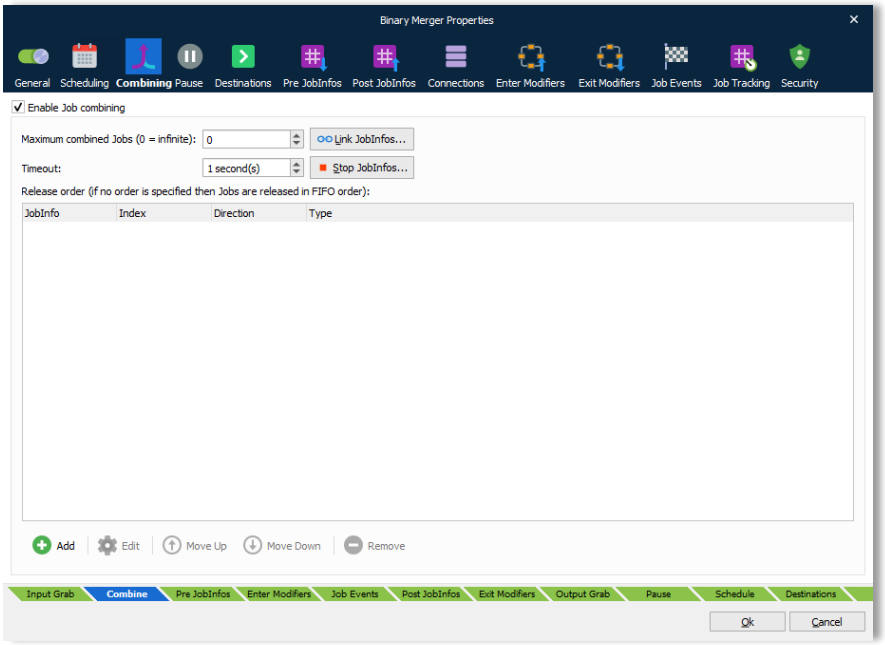
See also Binary Splitter.



The Binary Merger engine does not have any Setup tab.

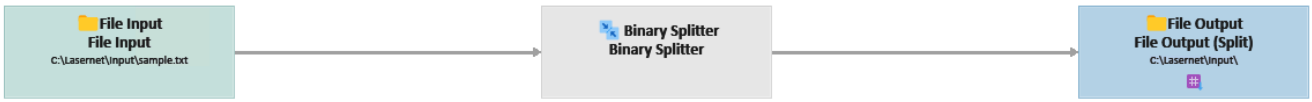
6.3.2.1 Combining

Select the **Combining** tab and activate **Enable Job combining** to merge binary jobs into a single job. More information about combiner settings can be found in chapter 12.1.

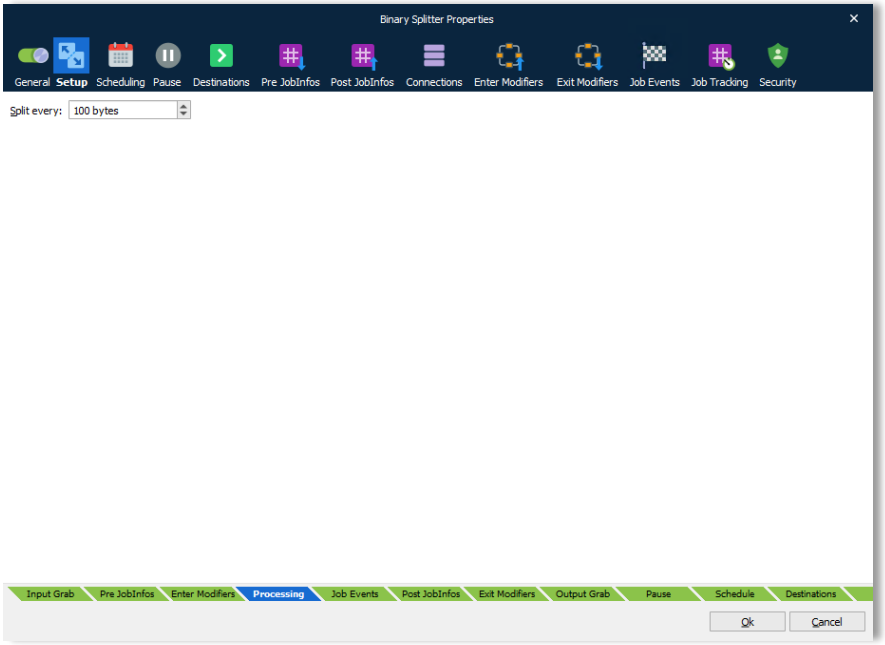


6.3.3 Binary Splitter

This engine splits a single text file, at pre-determined intervals, into chunks. See also Binary Merger.



The Binary Splitter engine has the following option:



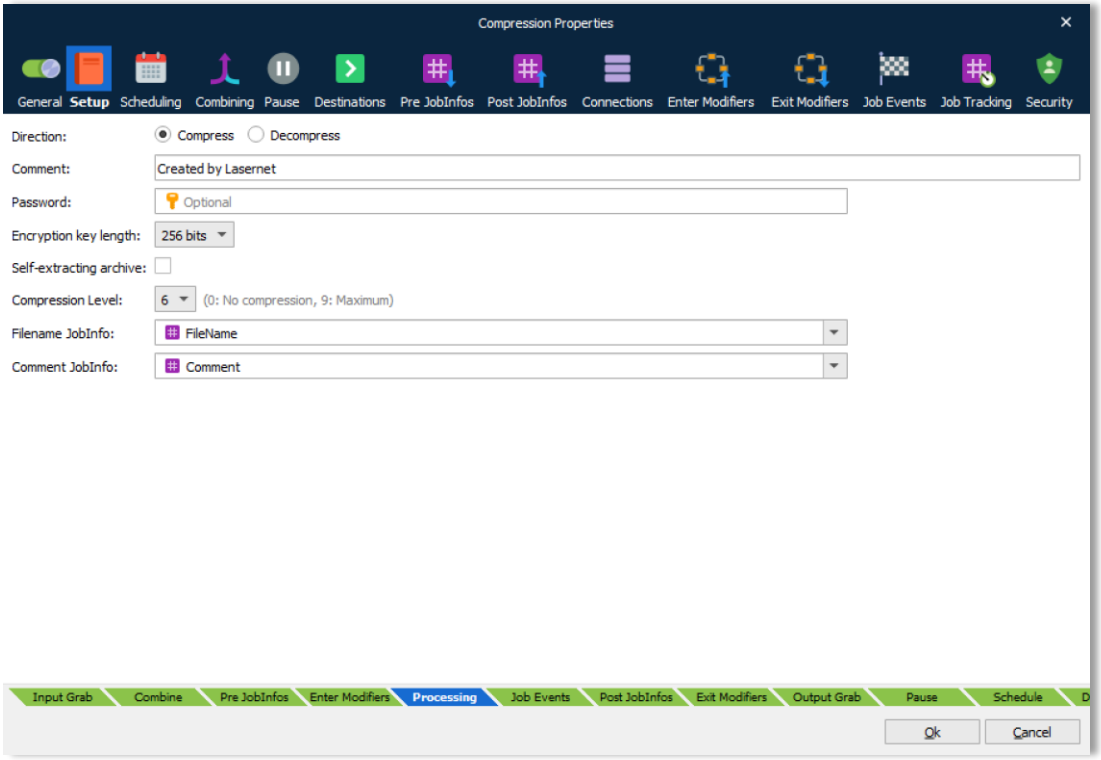
Split every – The interval at which the input file will be split. It is also the number of bytes each split document will contain.

6.3.4 Compression

The Compression Engine/Modifier is used for decompressing and compressing ZIP archives stored in JobData.

The engine is able to both decompress and compress. It is used when you have a ZIP archive containing more than one file for which you wish to generate Jobs.

The engine is also used when you have multiple Jobs which you wish to ZIP into one archive.



6.3.4.1 Properties

A range of properties can be set to control the resulting ZIP archive.

Password	If ZIP archive must be encrypted, a password is required.
Encryption key length	Length of the encryption key.
Self-extracting archive	Lasernet generates a self-extracting executable rather than a ZIP archive. Extension of the file will be set to ".exe".
Compression level	The higher the value, the smaller is the ZIP file. Higher values take a longer time, but should not be set lower unless a real performance problem exists. Can be overruled at runtime via JobInfo CompressionLevel.
Filename JobInfo	Name of file for ZIP archive.

JobInfo FullFilename must be specified. Paths are supported, excluding the drive letter.

6.3.4.2 Decompress

All files in the ZIP archive are extracted to individual Jobs. Password protected archives are supported.

A range of JobInfos are set on the created Jobs mimicking a File Input module:

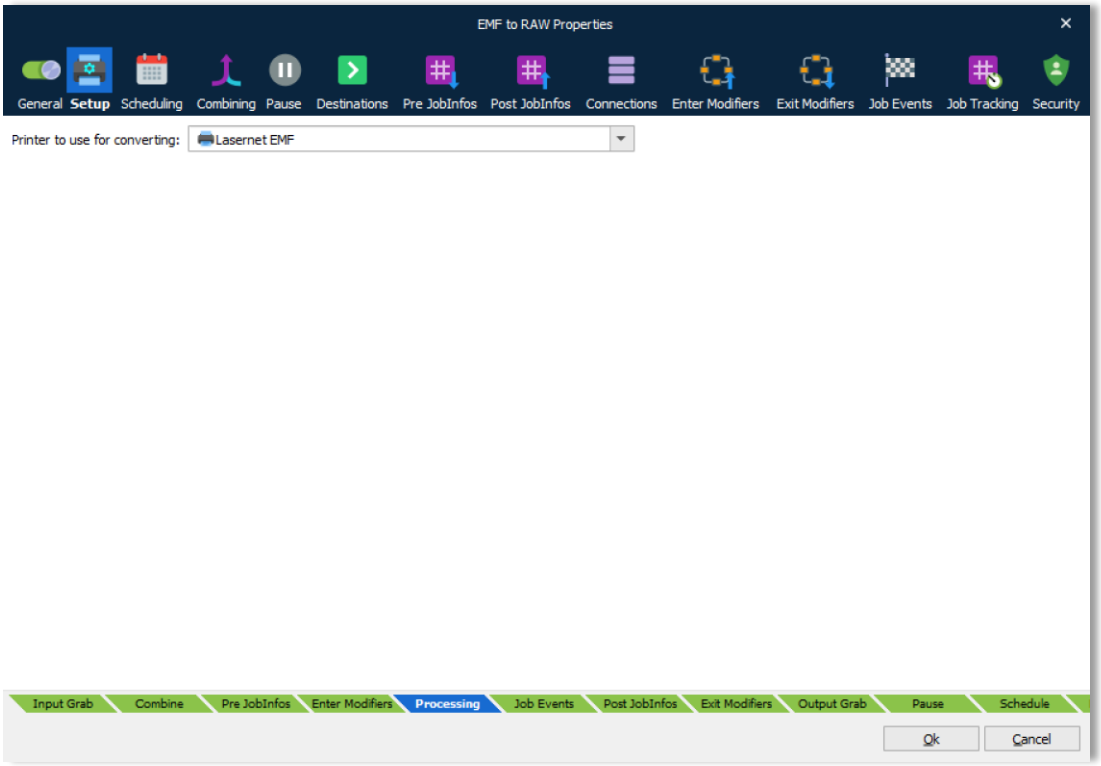
ArchiveFilename
CompressionLevel
FileLastModified
Filepath
Filename
FullFilename
Filesize
FilesizeCompressed
Extension
FilenameWithoutExt

6.3.4.3 Combining

Select the **Combining** tab and activate **Enable Job combining** to embed multiple jobs into a ZIP container. More information about combiner settings can be found in chapter 12.1.

6.3.5 EMF to RAW

Used for converting print to a specific printer's output format.

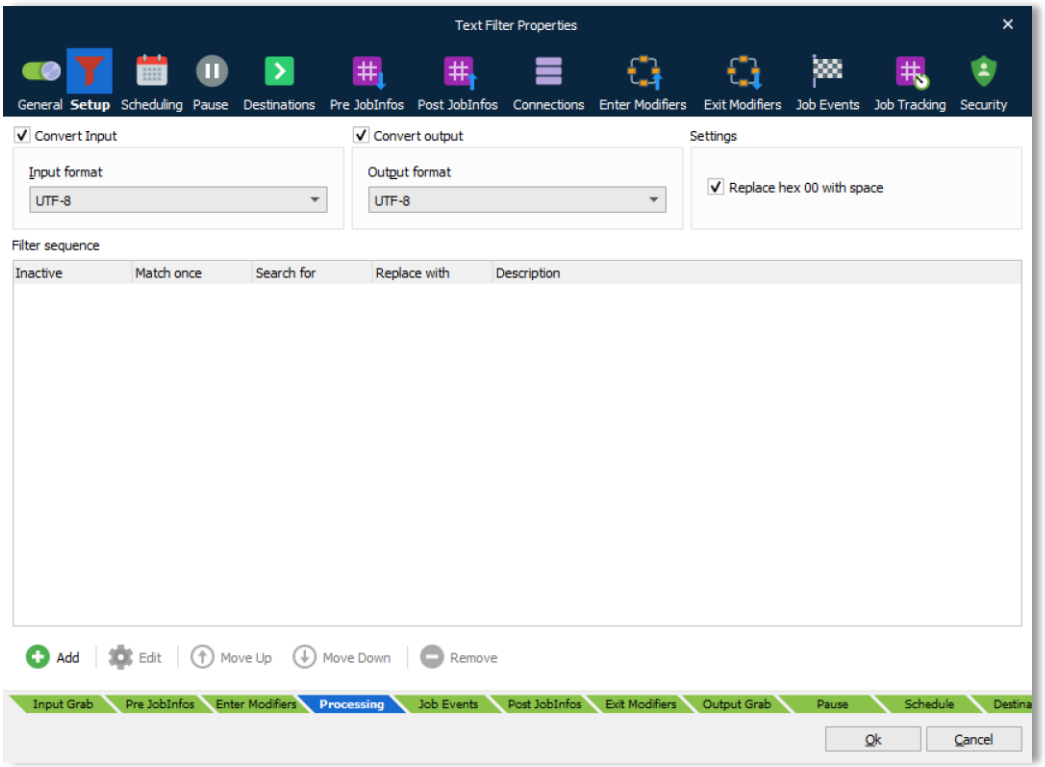


Use any local printer (recommended) or type a UNC path for a network printer to be used for converting EMF data into any RAW format created by the printer driver. Only true printer drivers are supported, as the physical driver creates the final result.

Printer drivers used for converting EMF into PDF, like drivers based on GhostScript and Adobe Distiller, are not supported since the output from those drivers is PostScript and thereafter a local DLL is used for converting PostScript into PDF.

6.3.6 Text Filter

The Text Filter is used for filtering the incoming job into a format that Lasernet can work with. When adding a new Text Filter engine, a dialog box for setting the character conversion is shown:



Replace hex 00 with space	Whether null bytes in the incoming data should be converted to spaces.
Input format	The set of characters which are used in the incoming data.
Output format	Always choose UTF-8 if the data should be processed by the Form Engine afterwards.

Lasernet sets the JobInfo ActiveCodePage to the name of the code page that is used for the output. If **Convert output** is not checked, Lasernet will use UTF8 by default for the output, since the Form Engine expects input in UTF8.

If **Convert input** is not checked, Lasernet will verify if the *ActiveCodePage* JobInfo exists. If it does, Lasernet will convert from that code page to UTF16 – which the filter uses internally. If it does not exist, Lasernet will assume that the code page is already UTF16.

6.3.6.1 Code pages

The most common input formats are: ISO 8859-1 to ISO 8859-14 (Windows Latin 1 1252 and various formats), IBM 850 / Windows CP850 (MS DOS codepage 850)

Lasernet supports a list of built-in code page conversion tables as well as having a “Regional and Language Option” (see Windows Control Panel) located on the client running Lasernet. If your character set is not shown you can add it to your list of code page conversion tables in this dialog box.

To solve some of the problems with handling various code pages (language-dependent character sets), Lasernet stores text internally as Unicode, which is a universal character set that will eventually contain all characters from all languages. It is therefore necessary to convert to and from the various code pages that exist around the world.

Lasernet uses the space efficient 'UTF-8' format to store Unicode text internally. UTF-8 is one of a number of ways of storing Unicode text. Lasernet comes with support for some code pages as well as supporting the code pages that are installed on the Windows system on which it is running.

It is always possible to convert from a code page to Unicode (since Unicode essentially includes all characters), but it is not always possible to do a perfect conversion from Unicode to any code page. This is because the Unicode text may contain characters that are not represented in that code page. Equally, converting between random code pages is not always successful as the same characters may not be present.

To convert from one code page to another code page Lasernet provides two options: The Filter Engine and the Code Page Conversion Modifier. If you do not want to filter anything you should use the Code Page Conversion Modifier.

Character codes

Characters are represented by *character codes*. Character codes are generated and stored when a user inputs a document. Single-Byte character sets (SCS) provide 256-character codes. This is an adequate number to encode most of the characters needed for Western Europe. For example, the Windows Extended ANSI character set contains 256 characters consisting of Latin letters, Arabic numerals, punctuation and drawing characters.

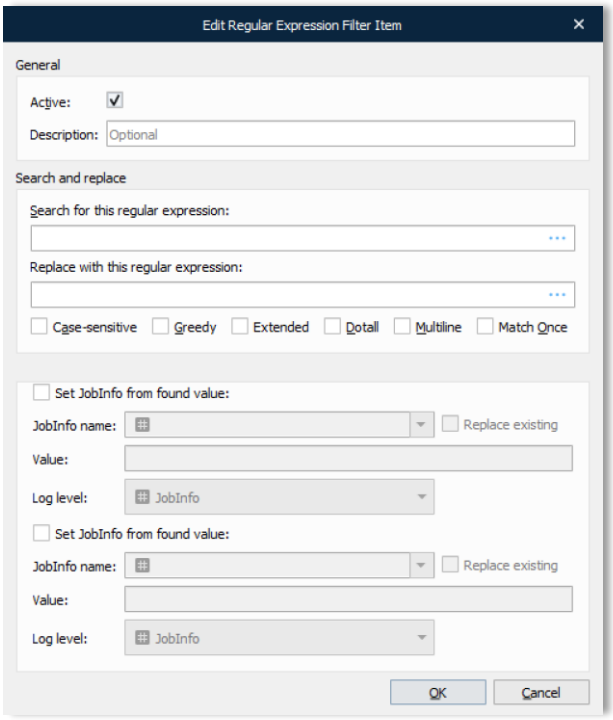
However, 256-character codes are not enough to represent all the characters needed by multi-lingual users in a single font, or by users in the Far East, where over 12,000 characters may need to be addressed at any time. Consequently, Multi-Byte character sets are necessary. Multi-Byte character sets (MACS) provide over 2 billion possible character codes (2 to the 31st power).

6.3.6.2 UTF8 Unicode

UTF8 Unicode is somewhat backwards compatible with the standard ANSI codepage. The first 192 characters are the same. This encoding allows files in standard ANSI to be read and written easily since they are more or less interchangeable (providing no characters above 192 are used).

6.3.6.3 Filter sequence

A filter definition can be defined for converting specific character strings in the job data. Lasernet provides pattern matching using regular expressions.



Search for this regular expression	The text to be searched for, typed as a regular expression.
Replace with this regular expression	The search value to be replaced, typed as a string or hex value. For example, \x0d\x0a.
Active	Turn on/off the chosen search/replace.
Greedy	Greedy makes the filter match as many items as possible. When it is turned off the filter matches as few items as possible. Relevant when using wildcards.
Multiline	Multiline causes ^ and $ to match before and after new lines instead of start and end of string.
Extended	Extended ignores white space characters in the expression – useful for formatting lone expressions.
Dotall	Dotall indicates whether, (dot) matches new-line characters.
Case sensitive	Enable case-sensitive matching.
Only match once	Only the first match will be replaced in job data.
Set JobInfo from found value	A unique search string can be put in to a specific JobInfo.

Set JobInfo from new value	A unique replace string can be put in to a specific JobInfo.

6.3.6.4 Capturing Text

The search expression allows for the use of parentheses to capture text for inclusion in the replace string. Consider this search string:

```
Chapter (\d.)
```

This searches for the text “Chapter” followed by a chapter number. The parentheses around the \d. are not searched for, instead the number found is remembered, and so it can be used in the replace string like this:

```
Chapter $1
```

The $ is followed by a number which refers to the sets of parentheses numbered from left to right. $0 is unique – it refers to the entire search string found.

It is possible to use captured text within the regular expression itself. To refer to the captured text, back references are used, which are indexed from 1. For example, duplicate words can be searched for in a string using **\b(\w+)\W+\1\b** this will match a word boundary, followed by one or more word characters, followed by one or more non-word characters, followed by the same text as the first parenthesized expression, followed by a word boundary.

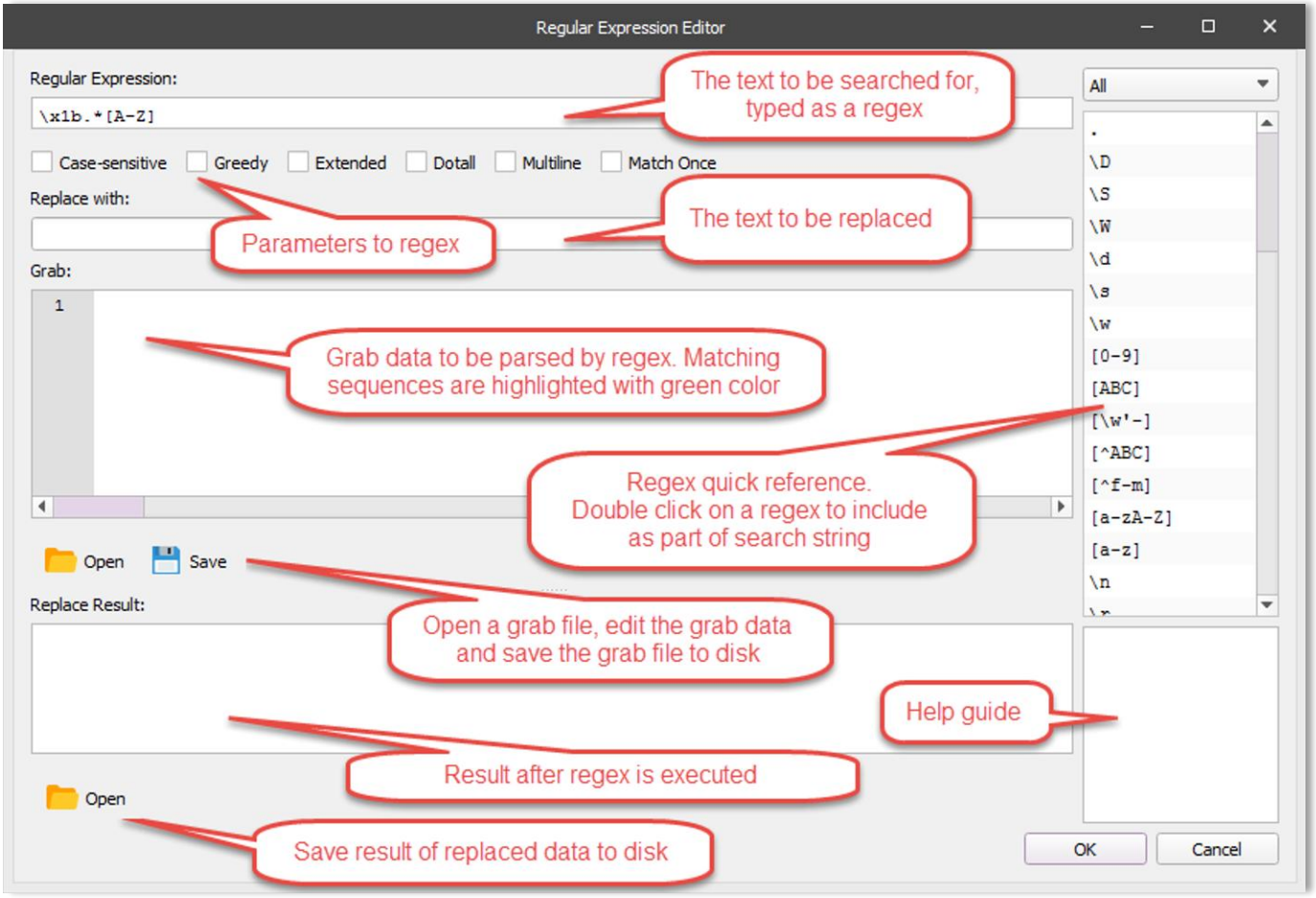
Parentheses can be used purely for grouping and not for capturing. This can be done with non-capturing syntax e.g., **(?:green|blue)**. Non-capturing parentheses begin '(?:' and end ')'. In this example either 'green' or 'blue' is matched but it is not captured, therefore it is only possibly knowing a match occurred rather than which color was actually found. Using non-capturing parentheses is more efficient than using capturing parentheses since the regular expression engine has to do less book-keeping.

Both capturing and non-capturing parentheses may be nested.

For more detailed information on regular expressions read the chapter about Regular Expressions.

6.3.6.5 Regular Expression Editor

The Lasernet Developer has a built-in regular expression editor for building and testing your regular expressions.



In the Regular Expression Editor, you can edit an expression and test it to see if it matches. It's a handy way to test regular expressions as you write them.

6.3.7 Form Engine

This chapter contains in-depth technical information about how the Form Engine works. For information about designing forms and rearranging text, see the Form Editor Guide on the Formpipe Knowledge Base.

Forms

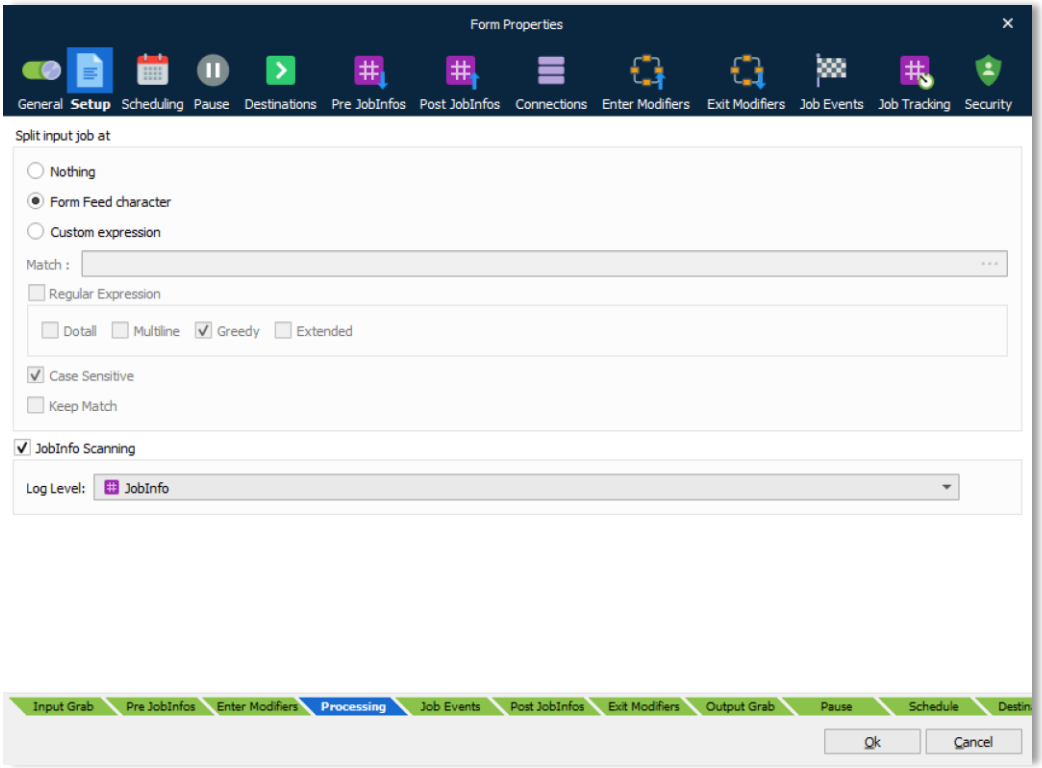
Each Form Engine has its own forms listed on the Forms tabs.

6.3.7.1 Entering the Form Engine

To make it possible to distinguish spool jobs from each other, the Form Engine adds a JobInfo SpoolJobID to the job the moment it arrives at the Form Engine. The SpoolJobID is unique and can be used for combining in the engines that follow, since it is automatically inherited by all derived jobs.

Since the Form Engine handles both XML and text input, it first checks whether the JobData is in XML format. It does this by trying to load the XML DOM. If it succeeds, then it continues by matching XML forms. If not, it will start to break the JobData into text pages. Additional input formats like CSV and XLSX are processed as XML.

NOTE: There is not a guaranteed one-to-one relationship between input pages and output pages. It depends on how the form is configured.



6.3.7.2 Text Pages

By default, the Form Engine assumes that each page is divided by a Form Feed (FF, ASCII 12, HEX 0C). To obtain this division use a Filter Engine or a Filter Modifier. The Form Engine searches through the JobData and for each Form Feed it finds it creates a Text Page that can be used in the Form Recognition process. It is configurable in the Form Engine properties.

Split input job at: You can customize the expression for which characters to match for splitting pages. This can be defined as a text string or a regular expression. Only one splitter is supported per Form Engine.

6.3.7.3 Line Printer compatibility

When building the text pages the Form Engine emulates some line printer facilities. It uses CR + LF to separate lines, but when CR and LF are presented individually they are treated as if it was a real line printer.

CR (Carriage Return) returns to the beginning of the line and starts “printing” on the same line again. In this situation spaces are ignored but new characters overwrite existing characters.

LF (Line Feed) moves the “print head” to the next line, but does not return to the beginning of the line.

```
This is a Test[LF]Next Line
```

Results in:

```
This is a Test
                    Next Line
```

Whereas

```
This is Test[CR]Next Line
```

Results in:

```
Next Linest
```

Finally:

```
This is a Test[CR][LF]Next Line
```

Results in:

```
This is Test
Next Line
```

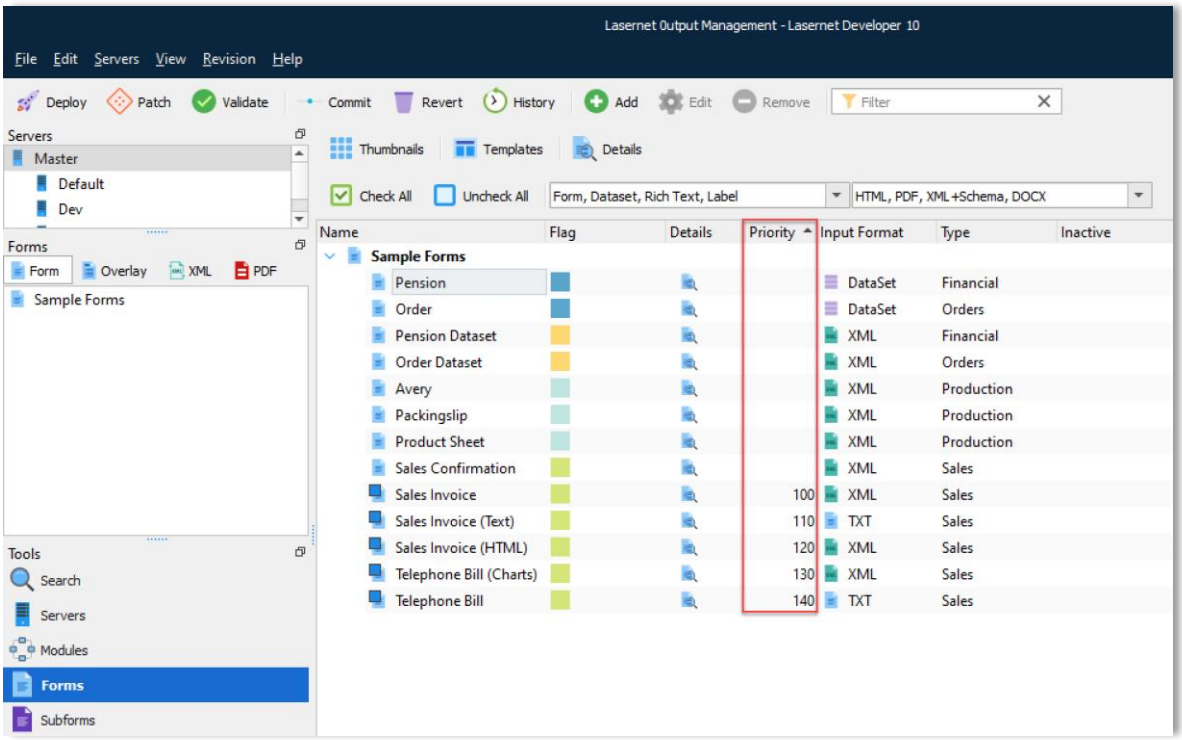
6.3.7.4 Form Recognition

Lasernet has a list of rules for how to recognize forms. As the basis for this process, to ensure that the appropriate form is recognized for each incoming job, add unique form criteria to each of the forms in a configuration. Alternatively, if form criteria cannot be unique, add a priority value to the properties of each form to enable Lasernet to select the appropriate form.

Note: Forms that do not have form criteria will never be recognized when Lasernet processes jobs.

The rules are similar for forms with XML and text as the input format:

- If no form is recognized for an incoming job, no form is processed.
- If multiple forms are recognized (according to their form criteria), the form with the lowest priority number is processed. If no priority is added to a form, the default priority of 0 (zero) is used.
- If multiple forms are recognized and have the same priority, the forms are prioritized alphabetically.



When all the incoming jobs are matched, the Form Engine loops through the pages and for each page determines which form it matches. If the form is not in combining mode, it is analysed and processed. When testing each page, a JobInfo PageWidth is set. This contains the number of characters in the widest line on the page. This can be used in the recognition.

When a form is recognized, two JobInfos are set: RecognizedForm, which contains the name of the form matched, and RecognizedSubjob, which contains the number of the sub jobs/pages in the entire job.

Combining input documents

To support the concatenation of several individual, but related documents, into one big document, the Form Engine uses the combining mode. This mode is configured separately for each form. The Form Engine combines individual documents (with the option of including their own header(s)/footer(s)) into one big document with (usually) one header/footer.

The Form Engine has several internal document buffers, one for each Form Engine/Input module combination. Each Form Engine (normally there is just one) has a single document buffer for each Input module. This buffer is used for storing the documents temporarily while collecting them.

The rules for concatenating the documents and entering them into the queue are somewhat complicated. This description will focus on the rules that make the Form Engine stop concatenating the documents and then processing them. Since only documents that directly follow each other are concatenated, the criteria for stopping proves more useful to study. Please remember that the rules for a document, are the rules that are defined on the form that the document matches.

Forms different rule

The Forms different rule is the most authoritative of the rules. It simply means that if the current document being processed is found to have another form, aside from the documents currently in the buffer (all jobs in the document buffer will always be recognized as the same form), Lasernet processes the document buffer

and continues to check the current documents against the Stop criteria. You should not fully rely on the Forms different rule. Use Stop or Link criteria instead.

Stop criteria
Stop criteria are preferred over link criteria since they do not need to rely on the time-out. (Recommended)

Stop criteria are used when the documents do not have much in common, but where the last document always contains something recognizable e.g., invoice totals.

When a stop criterion is fulfilled, it means that the current document is the last in the sequence and must be processed together with all the other documents in the corresponding document buffer.

In some situations, there might be both a Stop criterion and a Link criterion. In this situation, the Link criterion must also be fulfilled to have the current document processed together with the documents in the document buffer.

Link criteria
If a link criterion is specified, then two documents are concatenated if, and only if, the link criterion is fulfilled. Usually Link criteria will be invoice numbers, for example, that are common on the documents.

If a Link criterion is not fulfilled, Lasernet will process the documents in the buffer and check the current document against the stop criteria, if any.

Time out
The time out rule is a safety precaution. It is needed if there is no stop criterion, so that the buffer for the last job is emptied. If, for any reason, no other rules work, Lasernet will process the document buffer after a time-out (usually 10 seconds). This should not be relied upon for processing the document buffer. Always make sure that at least one of the other rules applies.

Rule's overview
This table shows the possible combinations of stop and link criteria and what actions Lasernet performs when they are available, fulfilled or not fulfilled. The position of the current document in the job is shown as it has a direct impact on the actions that are taken.

For document 1 there are not as many possibilities, since neither link criteria nor the forms different rule can be applied.

- N/A = Not available/defined
- N/C = Not checked in this combination
- - = Not fulfilled
- + = Fulfilled

In the first rule for document 1, the Forms different rule is not checked, the Stop criteria is not fulfilled and the Link criteria is not checked (as there are no other documents to check the Link criteria against). Because of this, the Form Engine simply enters document 1 into the document buffer and continues with the next document, if it exists.

In the first rule for document X, the forms are not different, the Stop criterion is not fulfilled and the Link criterion is not fulfilled. In this situation the Form Engine processes all the documents in the buffer and enters the current document (X) into the document buffer, where it will await further processing.

It should be noted that if the Form Engine runs out of documents the time-out facility allows it to process the buffer after a certain time – usually 10 seconds. This is a fail-safe function and should not be relied upon for processing the document buffer. The Forms different rule, stop criteria or link criteria should be used instead.

Document	Forms different	Stop criteria	Link criteria	Action	Resulting documents in buffer
1	N/C	-	N/C	Put 1 into buffer	1
	N/C	+	N/C	Process document	0
	N/C	N/A	N/C	Put 1 into buffer	1
X (2,3 etc.)	-	-	-	Process buffer. Put X into buffer	1 (X)
	-	-	+	Put X into buffer	X
	-	-	N/A	Put X into buffer	X
	-	+	-	Process buffer. Process document X.	0
	-	+	+	Put X into buffer. Process buffer.	0
	-	+	N/A	Put X into buffer. Process buffer.	0
	-	N/A	-	Process buffer. Put X into buffer	1 (X)
	-	N/A	+	Put X into buffer	X
	-	N/A	N/A	Put X into buffer	X
	+	-	N/C	Process buffer. Put X into buffer.	1 (X)
	+	+	N/C	Process buffer. Process document X.	0

The concatenation processes

Basic concatenation simply concatenates the data from each page, one after another and returns the result. However, when using the combining mode, it is also necessary to select whether the header and footer should be kept and in which lines the header stops and footer begins. The Keep Header setting has an effect on the middle and last pages only. The header on the first page is always kept. Correspondingly the Keep Footer setting has an effect only on the first and middle pages. The footer on the last page is always kept.

If you choose not to keep the header on the middle and last pages, the Form Engine cuts the number of lines given from the top of each page before concatenating. This is similar to when the footer on the first and middle pages are not kept. Here, the number of lines given is cut from the bottom of those pages before concatenating.

The end result is that the Form Engine now has one large text job ready for processing.

6.3.7.5 Processing the page

This description is for both job mode and combining mode.

The first step in processing a page is to add some JobInfos to describe where in the job it is located.

FirstPageInJob Set to 1 if the first page is the first page in the whole job. Otherwise, it is set to 0.

LastPageInJob Set to 1 if the last page is the last page in the whole job. Otherwise, it is set to 0.

PagesCombined In Page to Job mode how many pages have been concatenated. Otherwise set to 1.

The second step is to create the grab (internally) and add a JobInfo GrabWidth, which contains the number of characters in the widest line in the grab area. At this stage the Form Engine scans the page for #JobInfo name=value# entries. The key thing to remember here is that if the *name* of the JobInfo is *overlay* (case-insensitive) the *value* is considered as a name of an overlay.

Any Form Start Modifiers are then run. Finally, it saves the grab to a file, if selected. When saving the grab file, it sets two JobInfos: GrabFilename and GrabDirectory.

The Form Engine then moves on to processing the individual sheets. It starts with the first sheet (the leftmost one in the Form Editor) and continues through to the last. For each sheet, the Form Engine creates a copy (clone) of the current job, to make sure that the processing of each sheet does not affect the following ones.

For each sheet, three JobInfos are set: SheetName and Sheet, which both contain the name of the sheet being processed and finally DataFormat which contains the output type of the sheet. Known values for the DataFormat are: CSV, EDI, EMF, PDF, TIFF and XML.

After processing a sheet, the job that has been generated is passed to the engines that follow. Once all of the sheets have been processed, the Form End modifiers are run. It is important to note that the Form End modifiers cannot affect the job in any way. An example of Form End modifiers could be some custom statistics.

6.3.7.6 Processing a sheet

There are many different functions at work when processing a sheet. Most of the actions taken apply to both XML and text processing. There is little difference between XML and text input, but the output does differ somewhat.

Firstly, the early script objects are created and made available for the script. These are the grab and sheet objects. The scripting environment is then ready for the First pass scripts.

Next, the On Sheet Start Modifiers are run.

Then the actual analysis begins. All input patterns are found, both for XML and text. After the analysis, the Form Engine begins the Page Fitting routine for text output and XML generation for XML output. During this process, all First Pass scripts are run. When the fitting/generation has finished, the Late Script objects are created. They consist of all named rearranges as well as the pages array for the text output.

Then the On Before Recalcs modifiers are run. The scripting environment is made ready for the Second Pass scripts. The recalcs are then run. This is done in a different way for text and xml output. See the description below. After the recalcs, absolute rearranges are fitted for text output. XML output does not have absolute rearranges as such.

After all, recalcs have been calculated, the fitting process is done, the Form Engine generates the output in JobData and the After Analysis Modifiers are run. This means that the After Analysis modifiers can be used to replace JobData.

The final step in processing is running the On Sheet End Modifiers. The job is then passed on to the receivers.

Recalcs

Recalcs is short for recalculable and is a term used for rearranges that should be “recalculated” once the fitting process has finished. The primary reason for having recalcs is to make the calculation of subtotals and page x of y expressions possible. The execution of recalcs is interleaved with the execution of On Single/First/Middle/Last Page Start/End Modifiers.

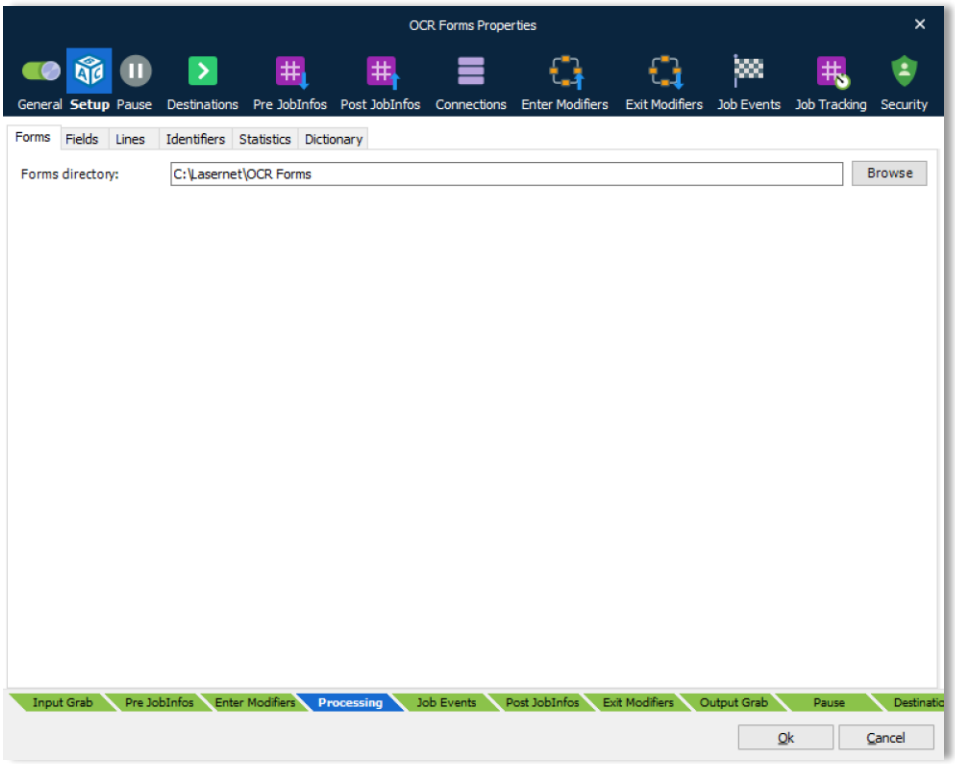
For each resulting page, the corresponding Page Start Modifiers are run. Then the recalcs for that page are executed and finally the Page End Modifiers. If there is only one resulting page, then only On Single Page Start/End Modifiers are run. For two pages, only First Page Start/End and Last Page Start/End are run. For three and more pages First/Middle/Last Modifiers are run.

The fitting processes

When fitting all the rearranges to result pages, the Form Engine follows a strict pattern. First it tries to see if it can fit everything onto a single page. If this does not succeed it moves on to a first page, which it fills with as much as it can. It then tries to see if the rest fits on a last page. If it does, the fitting process is finished. If not, it fills a middle page with as much as it can. It then tries again to see if the rest fits on a last page. If yes, the fitting process is finished. If not, it continues trying alternate middle and last page fittings until all rearranges have been fitted.

6.3.8 OCR

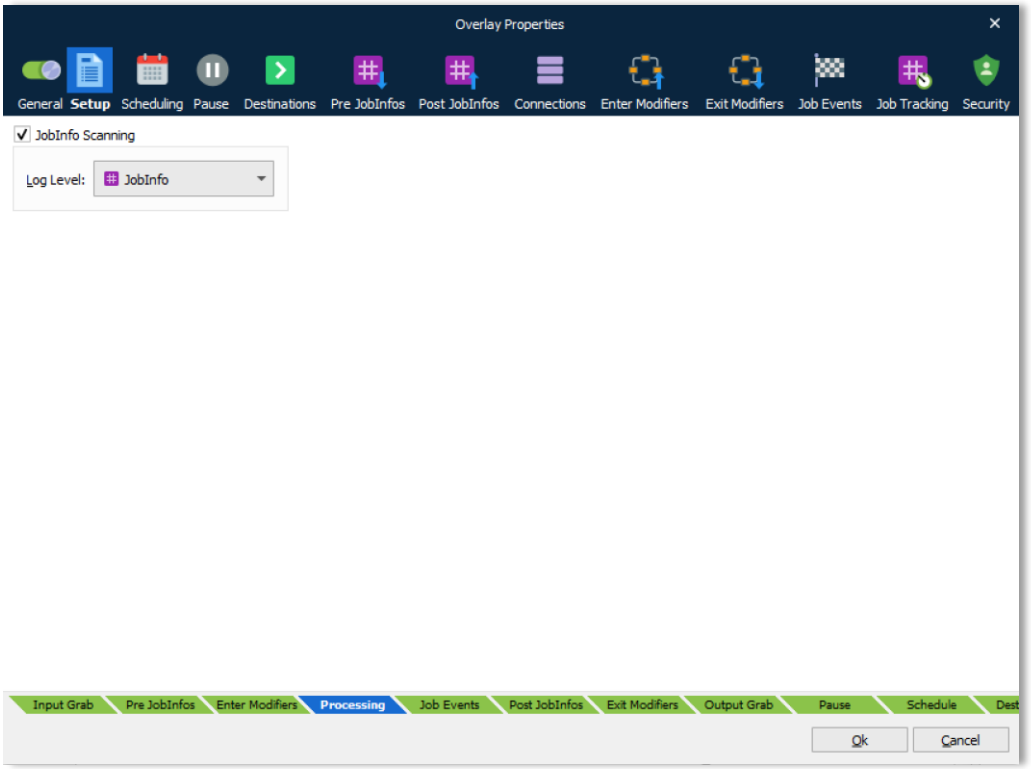
The OCR engine is a part of a client/server solution used for maintaining incoming business information and extracting data from TIFF and PDF documents.



For more information about how to setup the OCR engine, you need to install the Lasernet OCR application.

6.3.9 Overlay

This Overlay engine manages overlays.

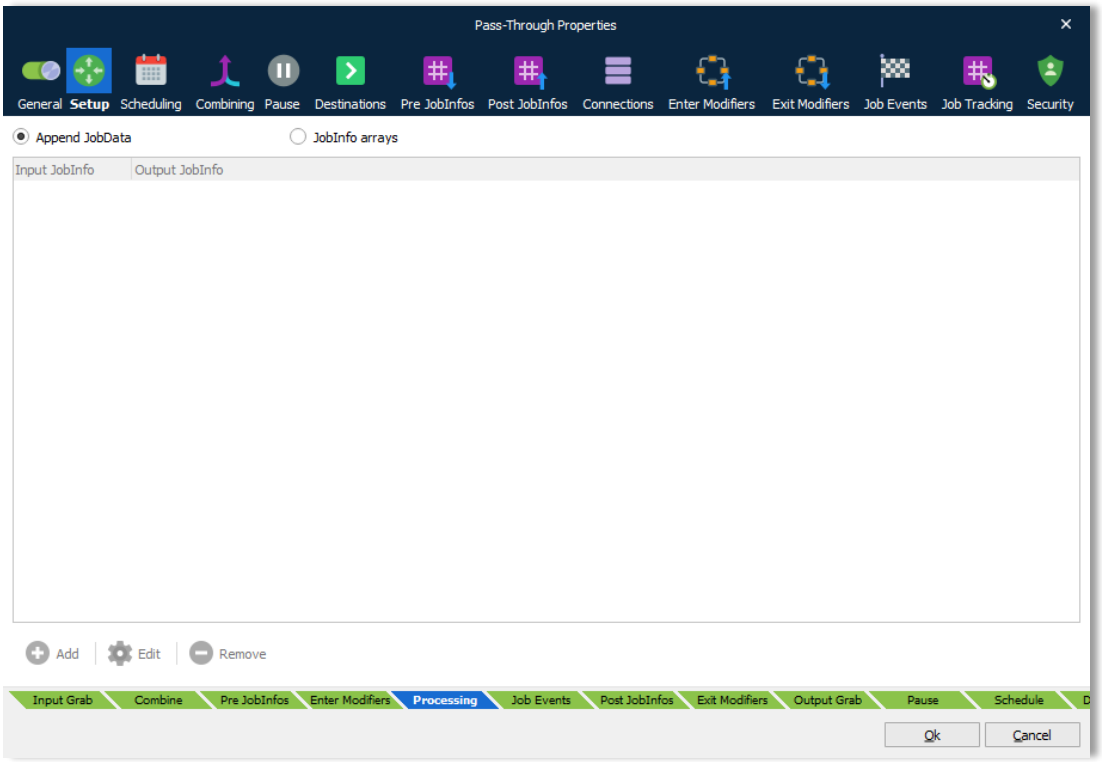


6.3.10 Pass-through

The Pass-through is a module that does not directly affect the job, but can be used for grouping a set of modifiers in a setup, as per the example below:

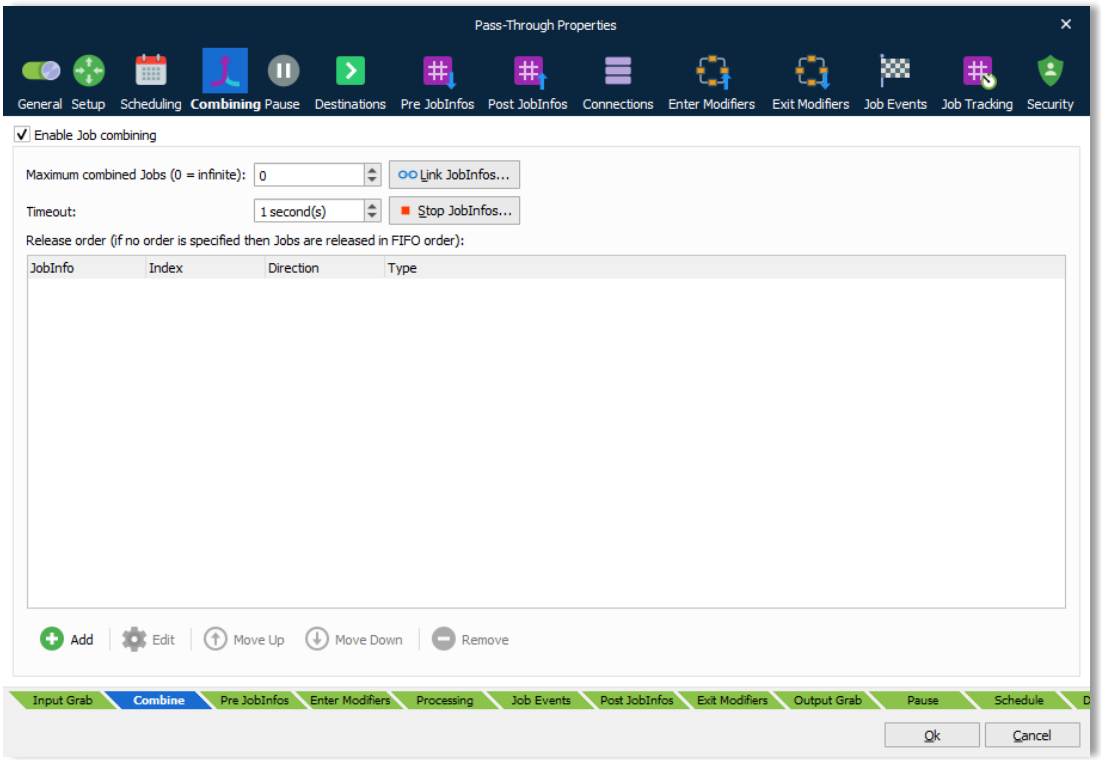
File Input A -> Pass-Through AB -> Forms -> File Output A

File Input B -> Pass-Through AB -> Forms -> File Output B



6.3.10.1 Setup tab

Settings defined in the Setup tab will only have an effect if Job Combining is activated in the Combining tab for the Pass-through module.



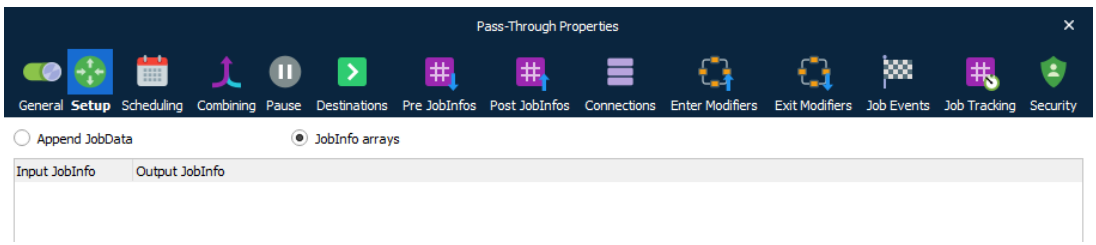
6.3.10.2 Append JobData

In the Setup tab, activate Append JobData to combine JobData and no other JobInfos (default mode). JobData will contain data for all combined jobs and only attach JobInfos available for the first job in the batch. JobInfos for all other jobs except the first job will automatically be deleted when combining is running. This is similar to how combining mode works in other modules (read chapter about combining mode).

6.3.10.3 JobInfo Arrays

Activate JobInfo Arrays if you want to create arrays for a list of user-defined JobInfos available in the batch job. Click the Add button in the Setup tab to add the name of a JobInfo (Input) which you want to copy into another JobInfo (output). The Output JobInfos will now contain an array of JobInfos that are attached to JobData.

If you want to create an array for an existing JobInfo, set Input JobInfo and Output JobInfo to the same name. If names for Input JobInfo and Output JobInfo are not the same, the value for Input JobInfo will be copied into a new Output JobInfo, which is added to the job. In both cases, a JobInfo array will be created for the Output JobInfo.



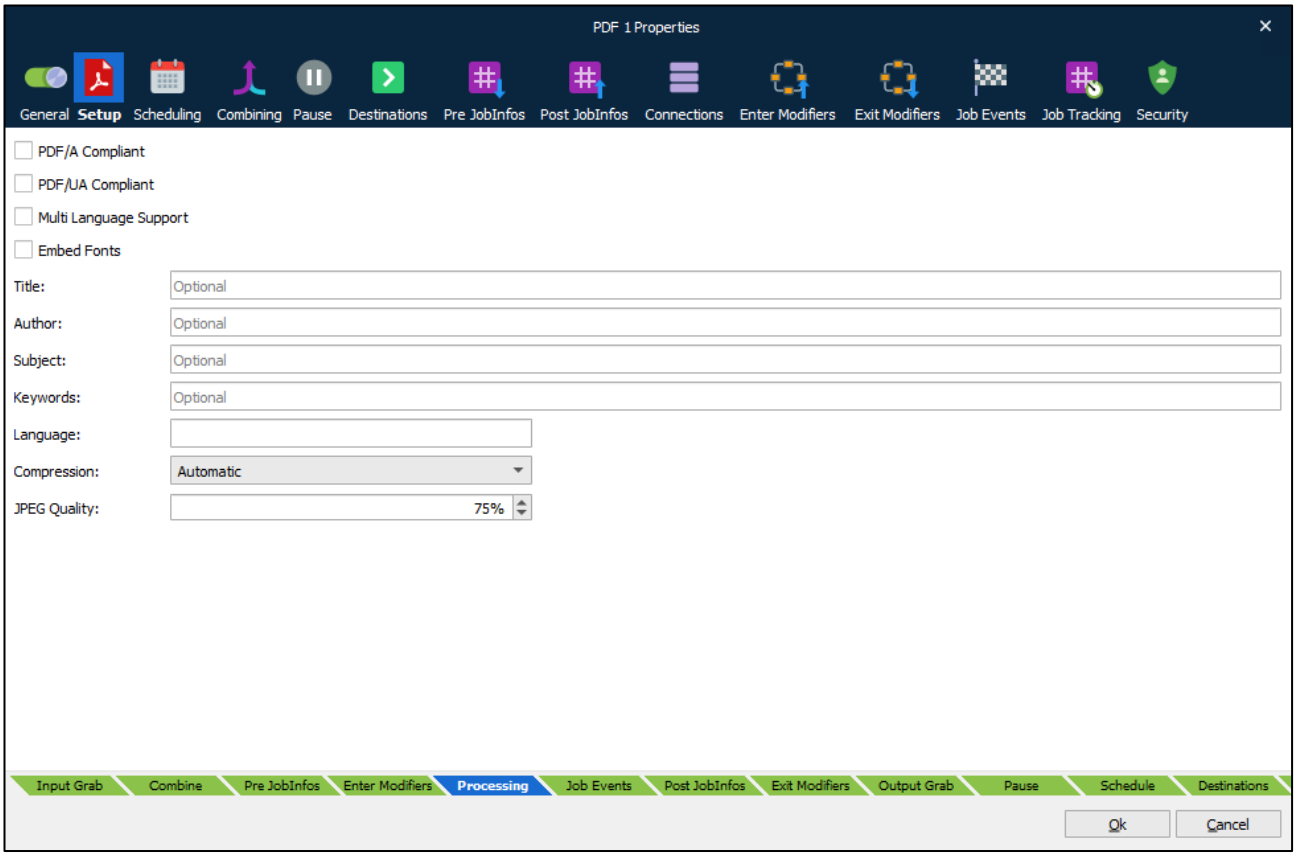
This feature is primarily used in setups where the destination in the Pass-through is a Web Service module and the result must contain a list of jobs with different filenames, content types etc.

6.3.11 PDF

The PDF Engine converts EMF data that is created by the Lasernet EMF printer driver, the Overlay Engine or the Form Engine, to PDF (reference version 1.7). The PDF Engine sets paper size and orientation for the document based on the paper formats included in the EMF data.

6.3.11.1 Settings

The PDF Engine has the following settings:



6.3.11.2 PDF/A Compliant

PDF/A is a PDF format typically used for the long-term archiving of electronic documents and is based on the PDF reference version 1.4. Fonts and color profiles will be embedded in the PDF file.

6.3.11.3 PDF/UA Compliant

The **PDF/UA Compliant** option is not applicable to the PDF engine. If you select this option, the PDF engine will not generate PDF/UA documents.

This option applies only to the Form engine (when it is selected as part of a sheet's configuration). For information about using this option with the Form engine, see *Configure PDF Output* in the Lasernet Form Editor Guide on the Formpipe Knowledge Base.

6.3.11.4 Multi Language Support

By default, the PDF format includes 7-bit ASCII characters only. Multi-language support must be activated for supporting and embedding additional characters and fonts.

6.3.11.5 Embed Fonts
Determines what fonts are embedded in the PDF file. Font embedding is used to ensure correct output on other client computers.

6.3.11.6 Title, Author, Subject, and Keywords
Descriptions to be included in the document properties of the PDF file.

6.3.11.7 Reference Printer
The physical name of a printer installed on the local computer (recommended) or on the network. The reference printer is used for rendering text on the output page. You should not pause this printer as it will stall all rendering. It is recommended that a new local printer be created as a reference.

6.3.11.8 Image Compression
Three options and a JPEG quality value (default 80%) are available for compressing images:

Automatic	Prefers indexed and uses JPEG if image contains more than 256 colors.
Indexed	Creates indexed images (using a palette) if the image contains no more than 256 colors. Otherwise stored as bitmap. The setting will give the best image quality but the file size may increase.
JPEG	Compresses all images using JPEG. The setting will compromise image quality for smaller file size.

6.3.11.9 PDF/A-3 - Tunneling Attachment(s) into PDF
The PDF engine and modifier have support for PDF/A-3 (ISO 19005-3:2012, PDF1.7). It allows embedding of various file formats (such as XML, CSV, word-processing documents, spreadsheet documents and others) into PDF/A compliant documents, plus additional PDF/A schemas in a Metadata section.

Embedding files in a PDF requires a set of JobInfos per file to be defined before PDF creation.

PDFEmbedFilename	Defines the name of the embedded file inside the PDF.
PDFEmbedRelationship	Set JobInfo to either *Source*, *Data*, *Alternative* or *Supplement*: The standard describes which value to use, depending on the embedded files relation to the PDF: **Source** - used if the file specification is the original source material for the associated content. **Data** - used if the file specification represents information used to derive a visual presentation – such as for a table or a graph. **Alternative** - used if this file specification is an alternative representation of content, for example audio. **Supplement** - used if this file specification represents a supplemental representation of the original source or data that may be more easily consumable (for example, a MathML version of an equation).

	If no PDFEmbedRelationship is defined, or it is set to an invalid value, the default value is *Supplement.*
PDFEmbedDescription	This JobInfo contains a description of the embedded file. The field is optional.
PDFEmbedSubType	This JobInfo must contain the MIME type of the embedded file. If not specified it will default to *application/octet-stream*.
PDFEmbedData	This JobInfo must contain the (binary) content of the file to embed into the PDF.

Any number of files can be embedded by creating an array of the JobInfos described above. The files are embedded at the PDF document level (it's currently not possible to embed files per page).

It is possible to embed files both when using regular PDF and PDF/A. If PDF/A is enabled, Lasernet will generate PDF/A-3B instead of PDF/A-1B.

6.3.11.10 Embedding PDF/A Extension Schemas and metadata in PDF/A.

You can embed additional PDF/A Extension Schemas and additional XMP Metadata in a PDF document. This is a part of the ZUGFeRD format used for electronic invoicing in Germany.

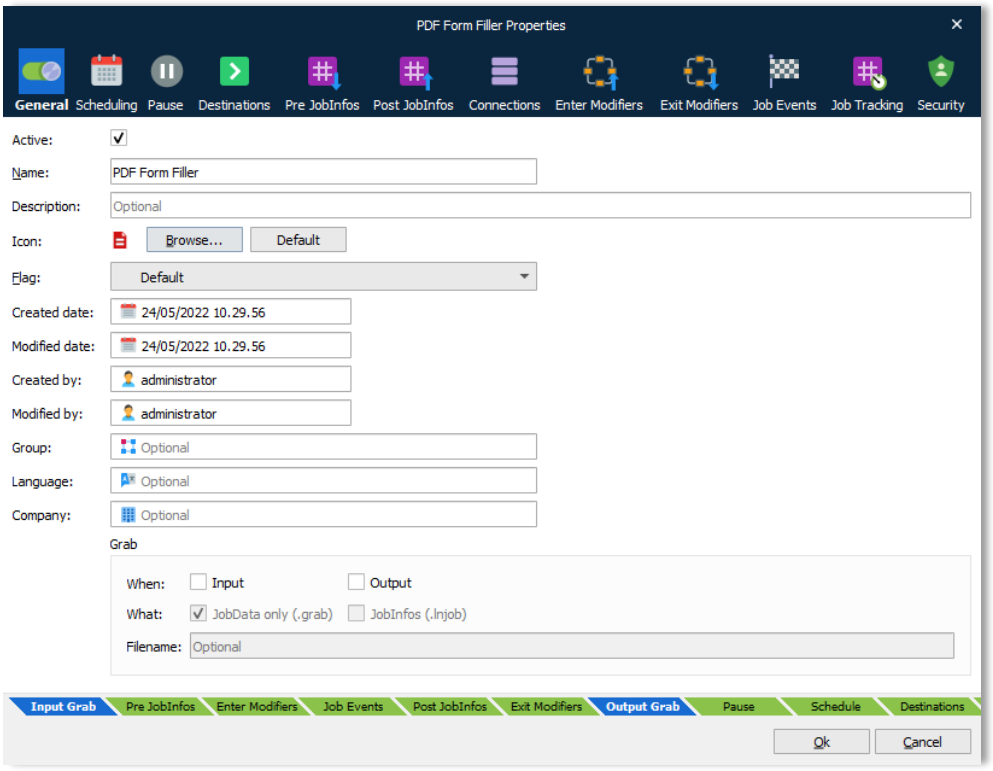
The **PDFEmbedPDFAExtensionSchema** JobInfo is used for PDF/A Extension Schemas. Any number of schemas can be embedded by creating an array. Each entry must be one or more valid rdf:Description XML element(s).

The **PDFEmbedAdditionalMetadataElement** JobInfo enables you to embed any number of additional XML metadata elements in the PDF by creating an array. Each entry must be one or more valid rdf:li XML elements.

XMP Metadata (and PDF/A extension schemas) are only included in the PDF when using PDF/A.

6.3.12 PDF Form Filler

The *PDF Form Filler* Engine is used to manage PDF Form Filler forms. The forms are designed to merge metadata with PDF filler fields created, for example, in Adobe Acrobat.



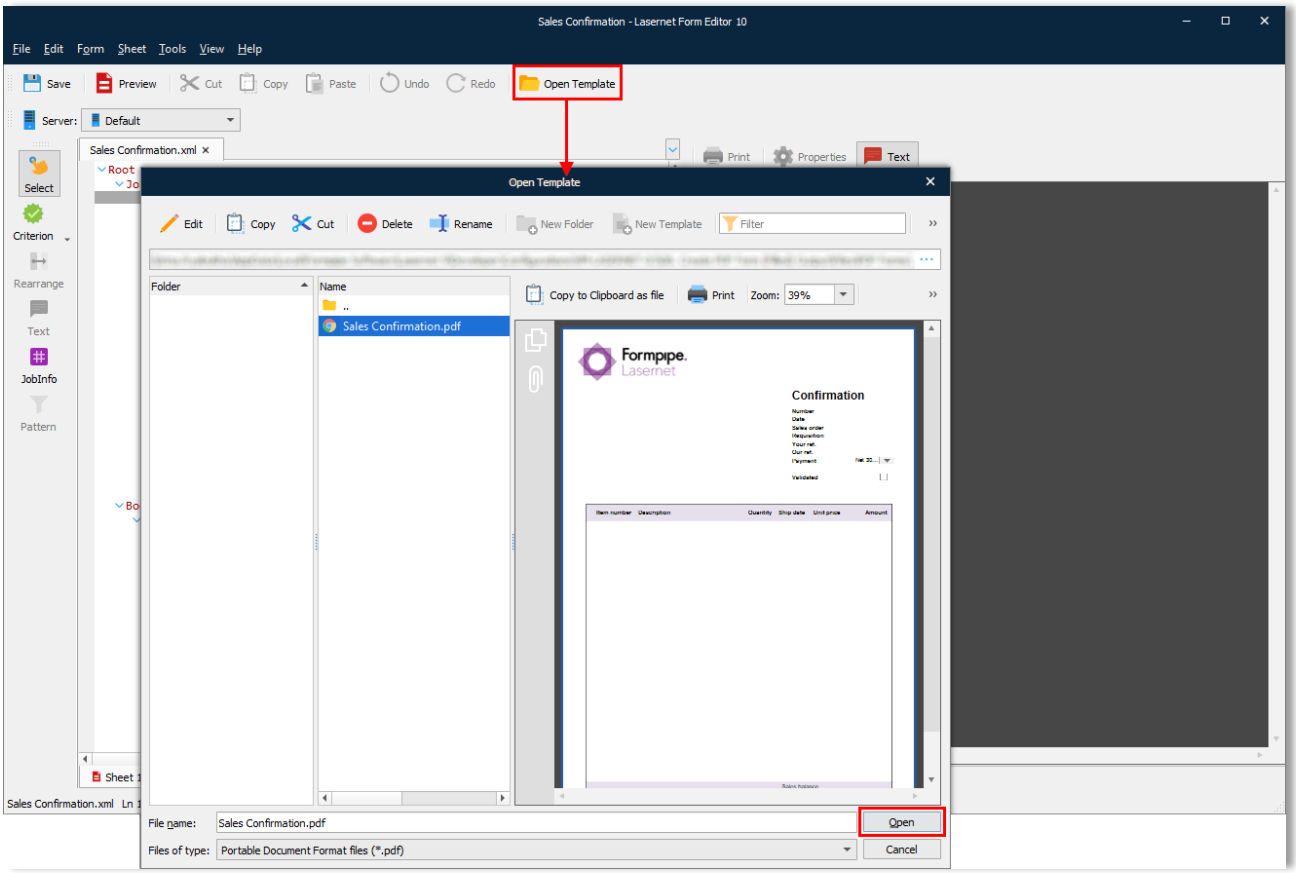
Each *PDF Form Filler* Engine has its own forms listed on the **Forms** tab.

6.3.12.1 PDF Form Filler Editor

PDF files may contain interactive elements such as text boxes, check boxes, combo boxes, etc. PDF Form Filler Editor is used for working with such PDF files.

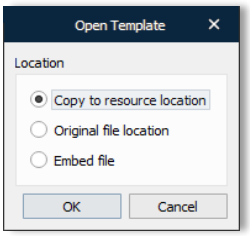
To add values for interactive fields available in the PDF file, follow the steps listed below:

1. Create a form and open it for editing.
2. In the form that opens, click the **Open Template** button to open a template.

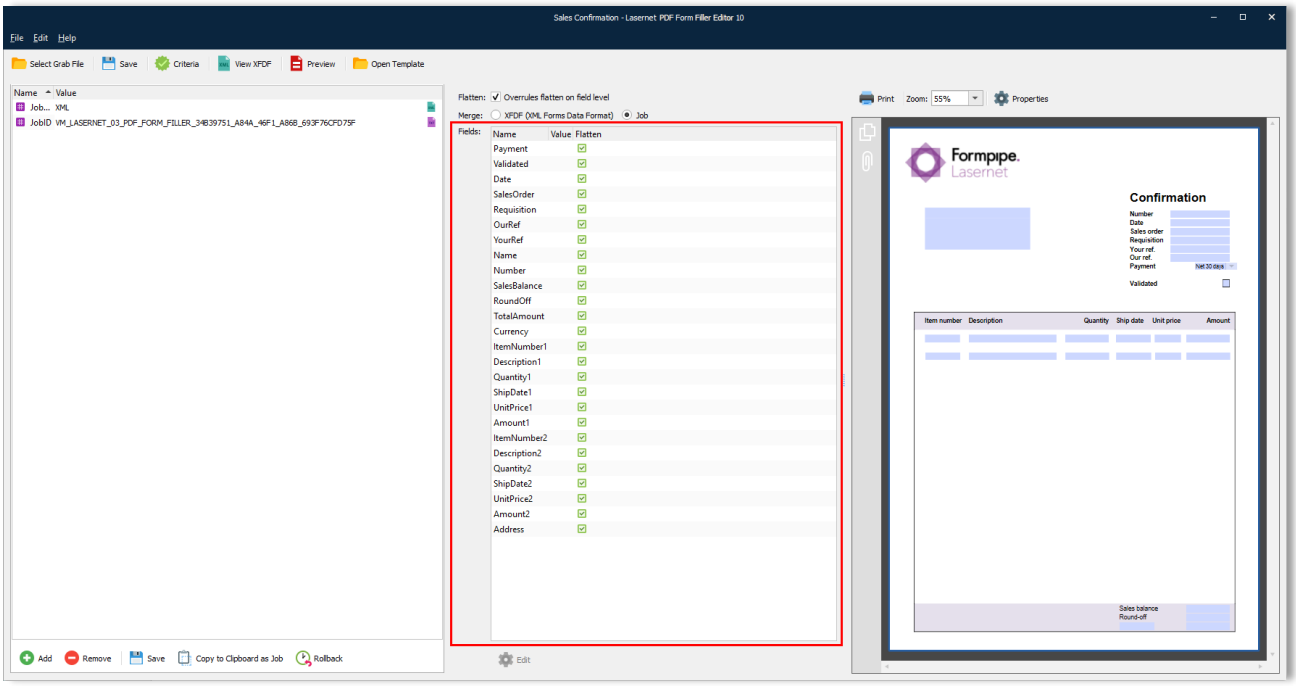


3. Once you click the **Open** button in the **Open Template** dialog box, you are prompted to select whether to embed a file into the form or just to refer to that file:

- **Copy to resource location**: allows copying a file you select to the **PDF Forms** folder of the Configuration Resources location. As a result, the form refers to the file which has been copied to the resource location.
- **Original file location**: allows the form referring to the selected file directly on disk, making it possible to change that file on disk without having to open a configuration, load a new template, as well as commit and deploy. If any changes to the template file imply adding interactive elements which are supposed to be used in Lasernet, the template shall be re-open.
- **Embed file**: allows using a template embedded with the form.

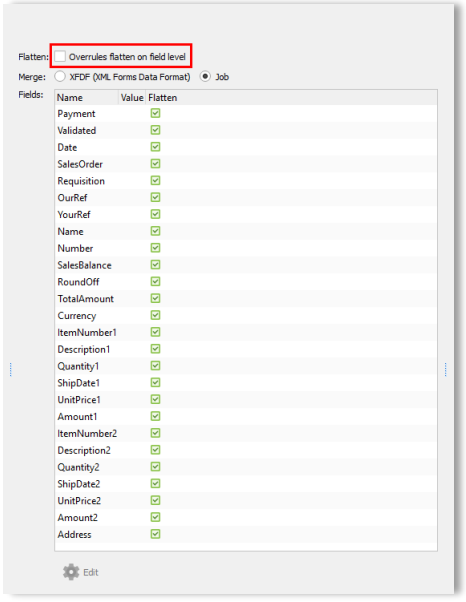


4. Once you open a template, a list of fields available in the template is automatically generated on the **Fields** pane in the PDF Form Filler Editor.

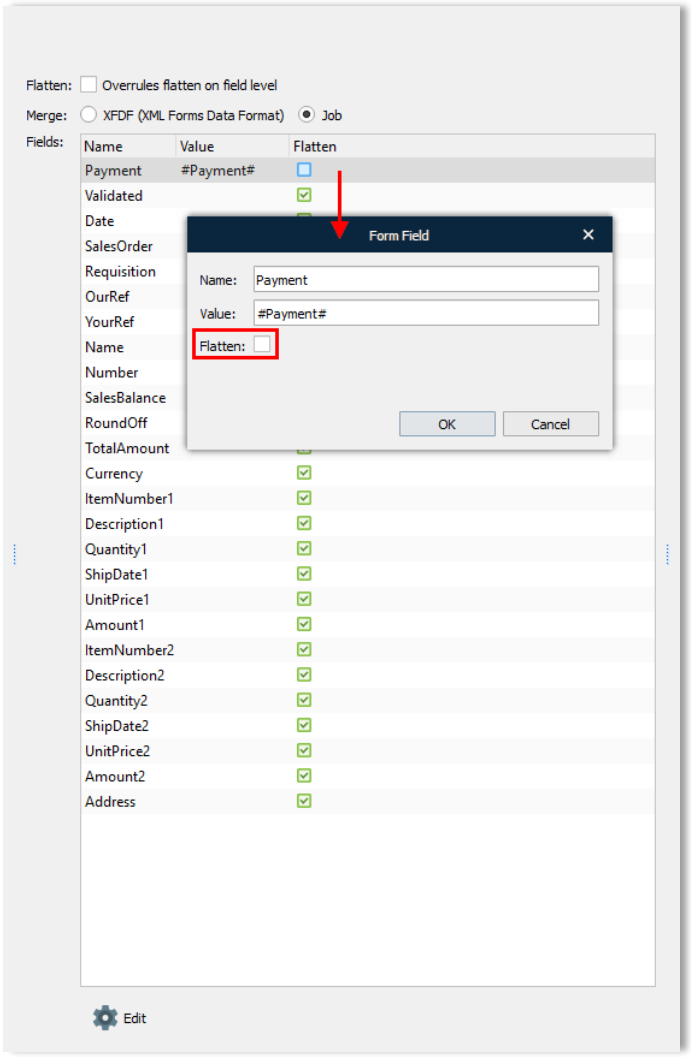


By default, all the fields are added as flatten meaning that a generated document contains fields filled with corresponding values, but it is locked for changing those values. However, you can disable the **Flatten** option to make interactive fields editable. To this end, follow the steps listed below:

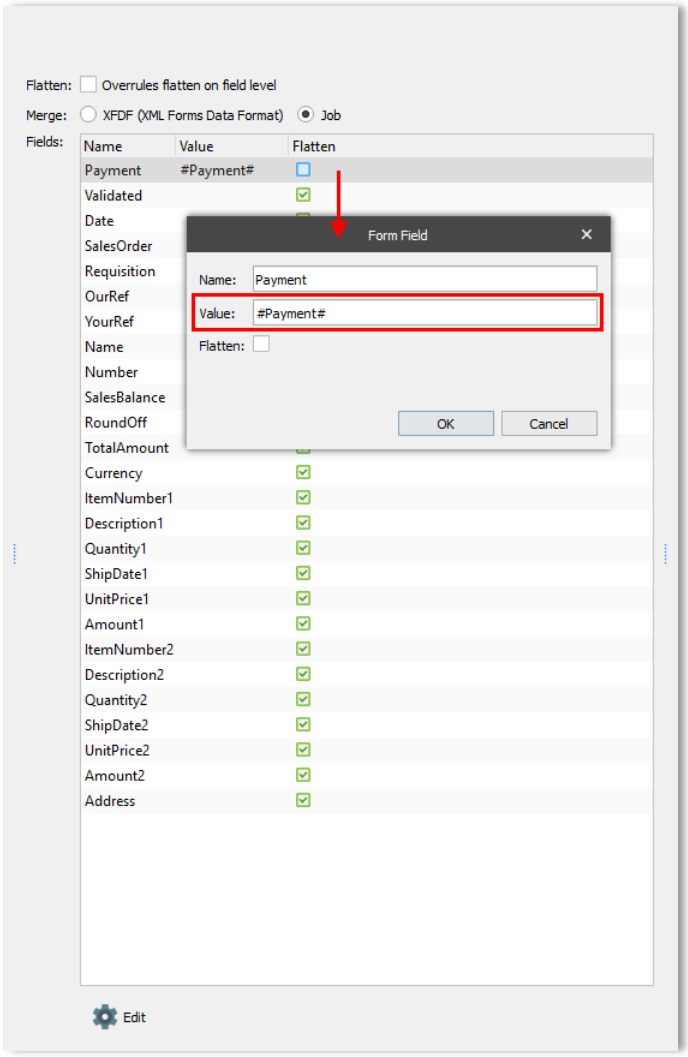
5. Clear the **Overrules flatten on fields level** check box. This enables manipulating the fields.



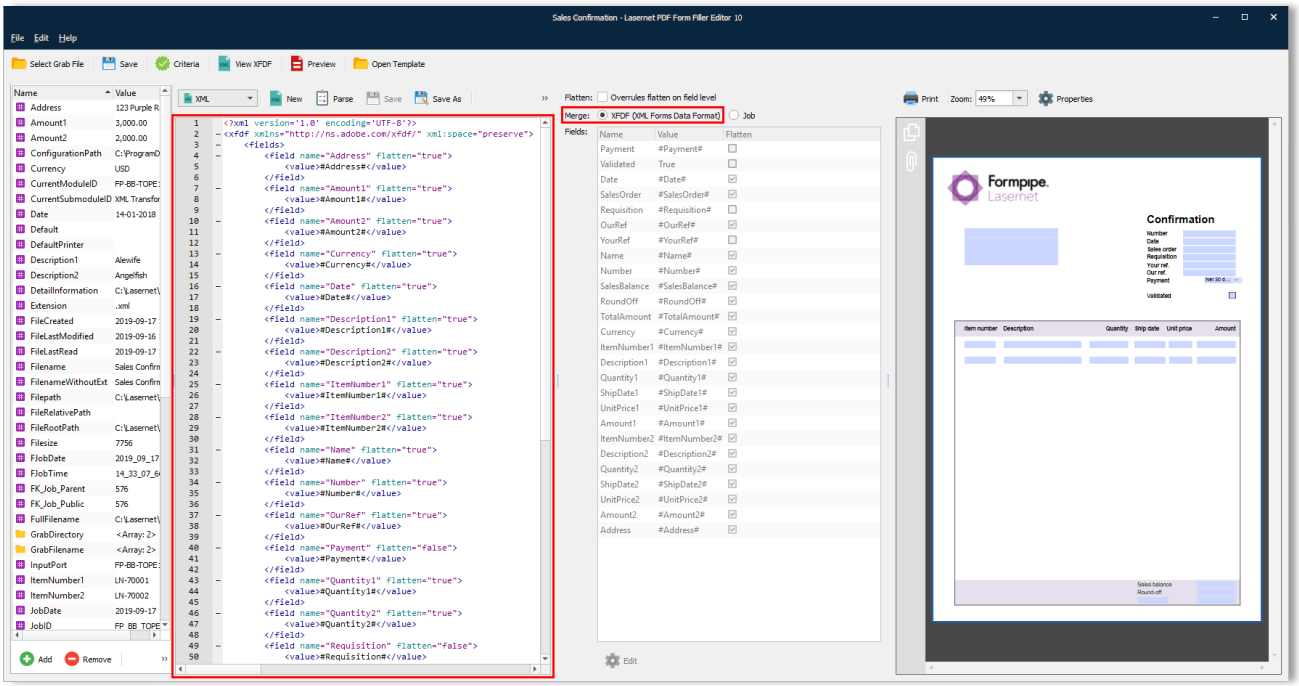
6. Additionally, you can clear the **Flatten** check box for that field which you want to unlock for manipulating. To this end, select the field record in the list, and then click the **Edit** button, or double-click the field record. In the **Form Field** dialog that appears, clear the **Flatten** check box.



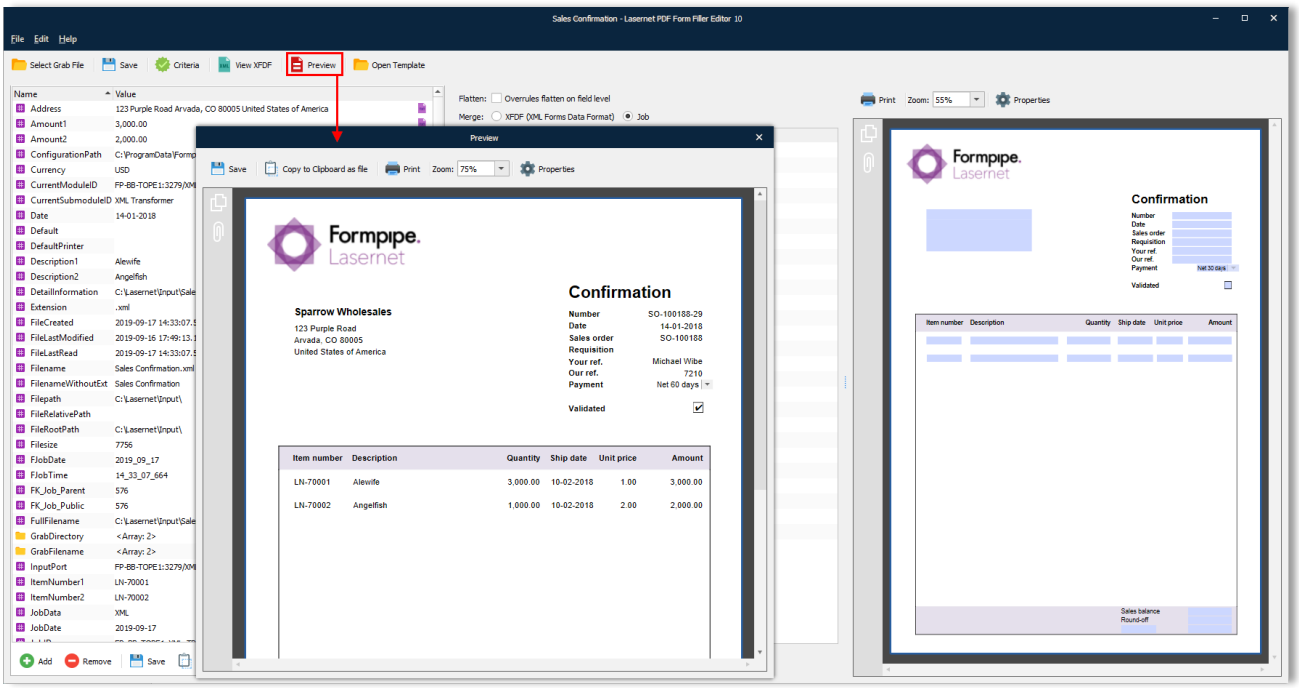
7. Use the **Form Field** dialog, to add a value to a field. To this end, specify the required value in the **Value** text box. You can define any JobInfo value using the JobInfo substitution mechanism (e.g.: *#Payment#*).



By default, all available interactive fields are merged as a job (the **Job** radio button is selected under **Merge**). For this mode, you are supposed to add values and flatten type (true/false) of each field manually. If you have the XFDF (XML Form Data Format) file, select the **XFDF (XML Form Data Format)** radio button under **Merge** and open the file by clicking the **Select Grab File** button. For this mode, field values as well as a flatten type (true/false) of each field are taken from this file.



8. Once you are done, you can preview the final version of your document containing fields with specified values by clicking the **Preview** button.

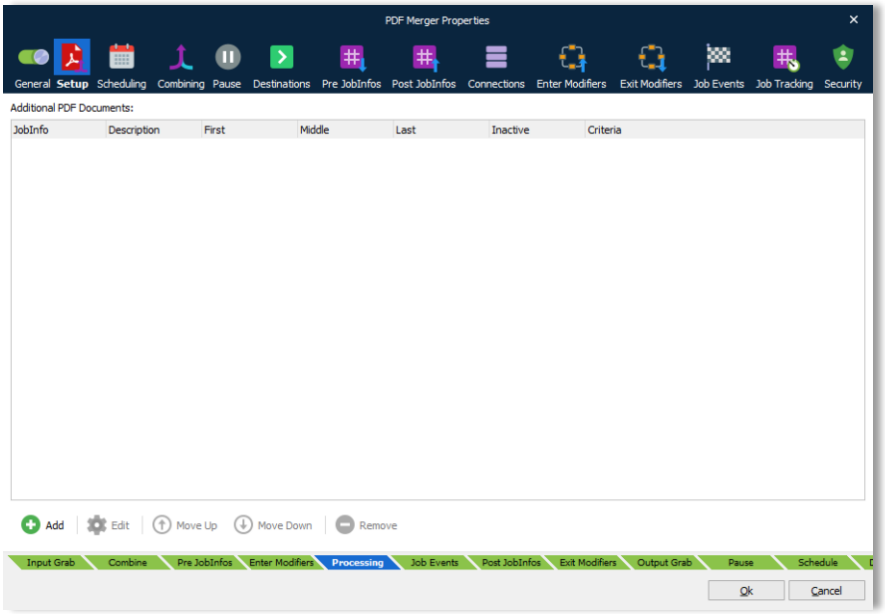


6.3.13 PDF Merger

A module to combine multiple PDF documents to a single job, as well as merge a list of additional PDF documents, with the combined job.

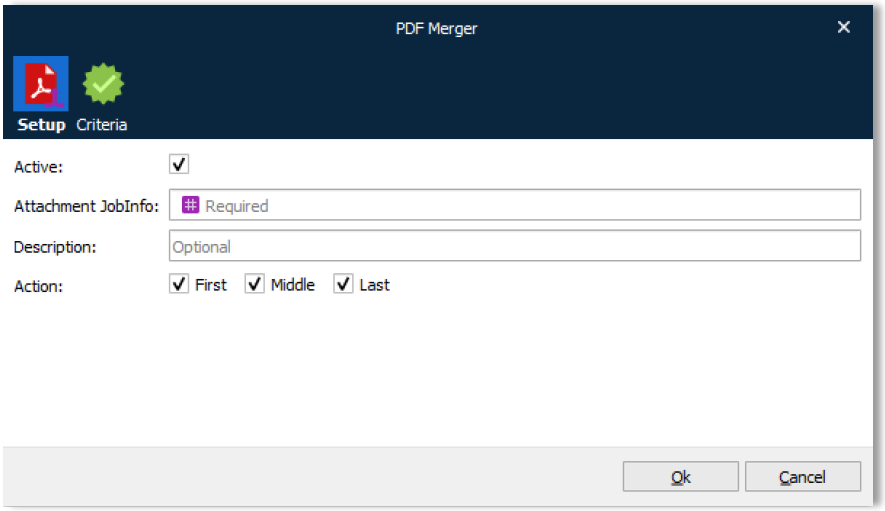
The JobInfo named 'JobData' will contain the data for all the combined jobs. It will be attached to the JobInfos that were created in the first job. JobInfos for all other Jobs expect the first job will be deleted.

You can add a criterion to each additional document as well as documents whose criterion are true, to be merged with the job. If no additional documents are added, it will only combine jobs from the combiner.



Settings

The PDF Merger has the following settings:



6.3.13.1 Active

Turn item on/off.

6.3.13.2 Attachment JobInfo
Name of the JobInfo that contains PDF data to insert at the specified location(s). PDF document data is read from the first job.

6.3.13.3 Description
User defined description field.

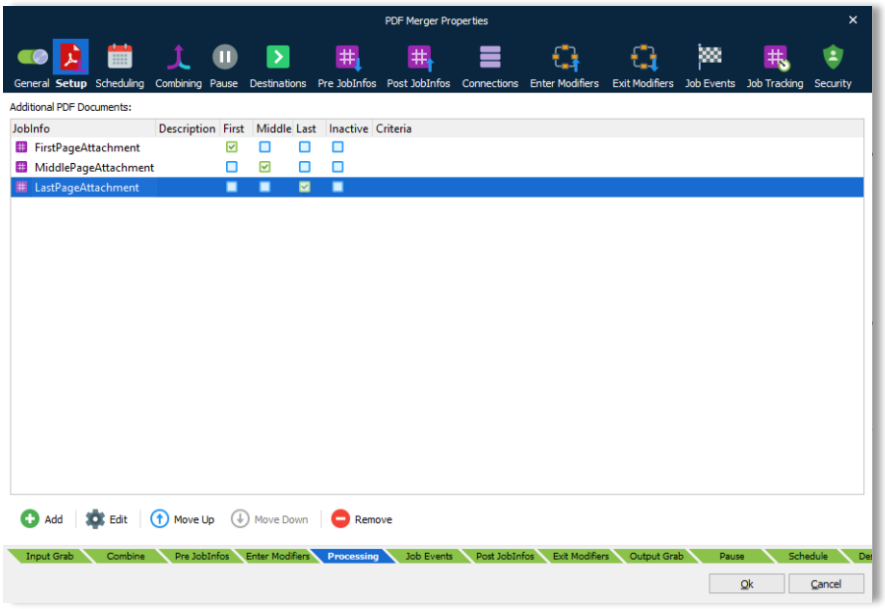
6.3.13.4 Action
Select where to insert an attachment in JobData.

First: Before first job

Middle: Between jobs

Last: After the last job

6.3.13.5 Non-valid/Existing PDF Data
If non-valid or existing PDF data is added in JobInfos listed to be merged with JobData at First, Middle and Last page, the engine will behave as follows:

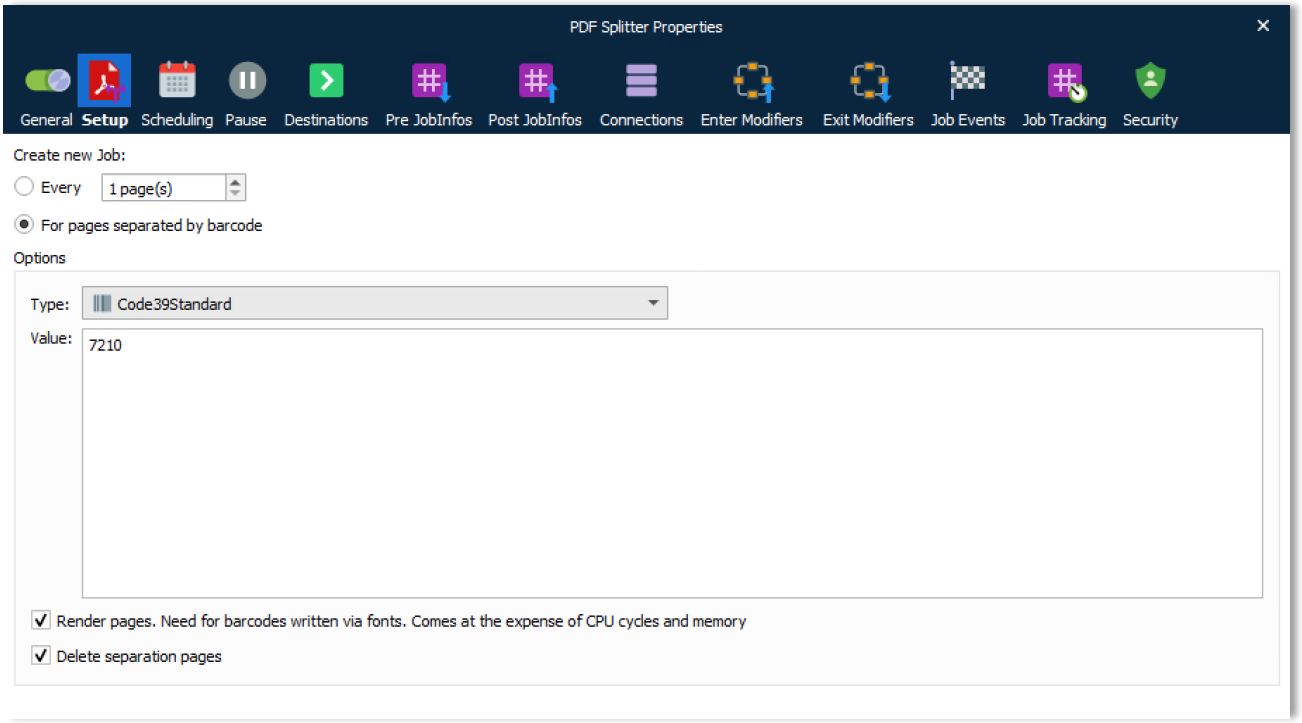


- If the listed JobInfo exists in the job but contains non-valid PDF data, the engine will fail the job.
- If the listed JobInfo exists in the job but contains null data, the engine will not fail the job.
- If the listed JobInfo does not exist in the job, the engine will not fail the job

6.3.13.6 Combining
Select the **Combining** tab and activate **Enable Job combining** to merge multiple PDF documents into a single PDF document. More information about combiner settings can be found in chapter 12.1.

6.3.14 PDF Splitter

Splits PDF documents into multiple parts.



6.3.14.1 Create new Job

Select the method by which you want Lasernet to determine when to split documents:

- **Every** – The PDF will be split at user-defined intervals. Adjust the number of pages between splits.
- **For pages separated by barcode** – The PDF will be split wherever there is a page containing a specified barcode.

6.3.14.2 Options

The following options are relevant to the **For pages separated by barcode** option.

- **Type** – Select, from the drop-down menu, the barcode type for which you want to search.

  Note: We recommend selecting a particular barcode type instead of All types because PDF Splitter will process the document faster.

- **Value** – Type the barcode value for which you want to search.

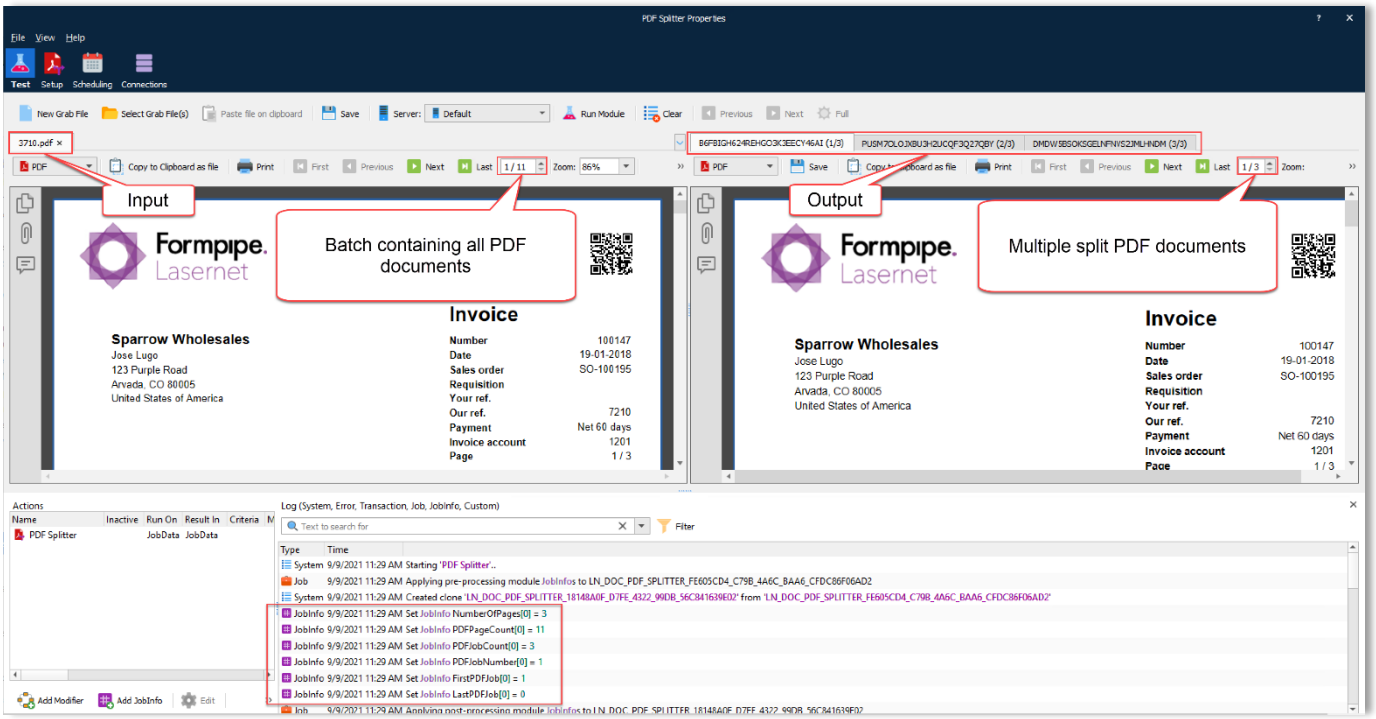
  Note: This is an optional field. If you do not specify a value, the engine will search for any value and might take longer to process the document.

- **Render pages** – If checked, barcodes embedded as fonts (vector graphics) will be recognised and processed.

  Note: Only select this option if your document contains font barcodes instead of image barcodes. CPU and memory load will increase, resulting in longer processing time.

- **Delete separation pages** – If checked, pages used to split the PDF will be automatically deleted from the output file.

The example shows one input PDF (batch) being split into three output PDF documents.



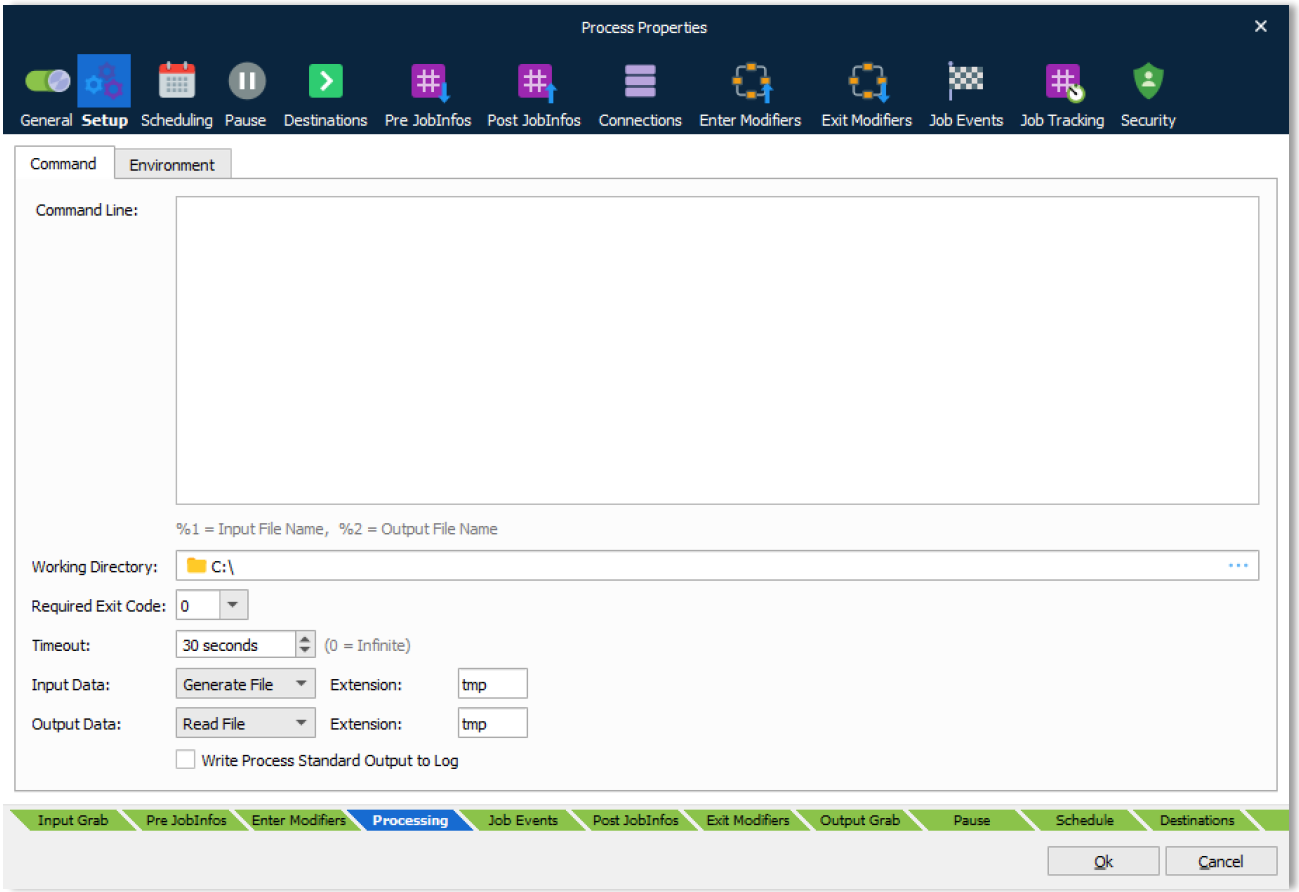
6.3.14.3 JobInfos

The following JobInfos are created by the PDF Splitter:

NumberOfPages	Number of pages in the current split job.
PDFPageCount	Total number of pages in incoming PDF job
PDFJobCount	Total number of split PDF jobs
PDFJobNumber	Job number for split PDF job
FirstPDFJob	Boolean set to 1 (true) when the first split PDF job in the batch is created
LastPDFJob	Boolean set to 1 (true) when the last split PDF job in the batch is created

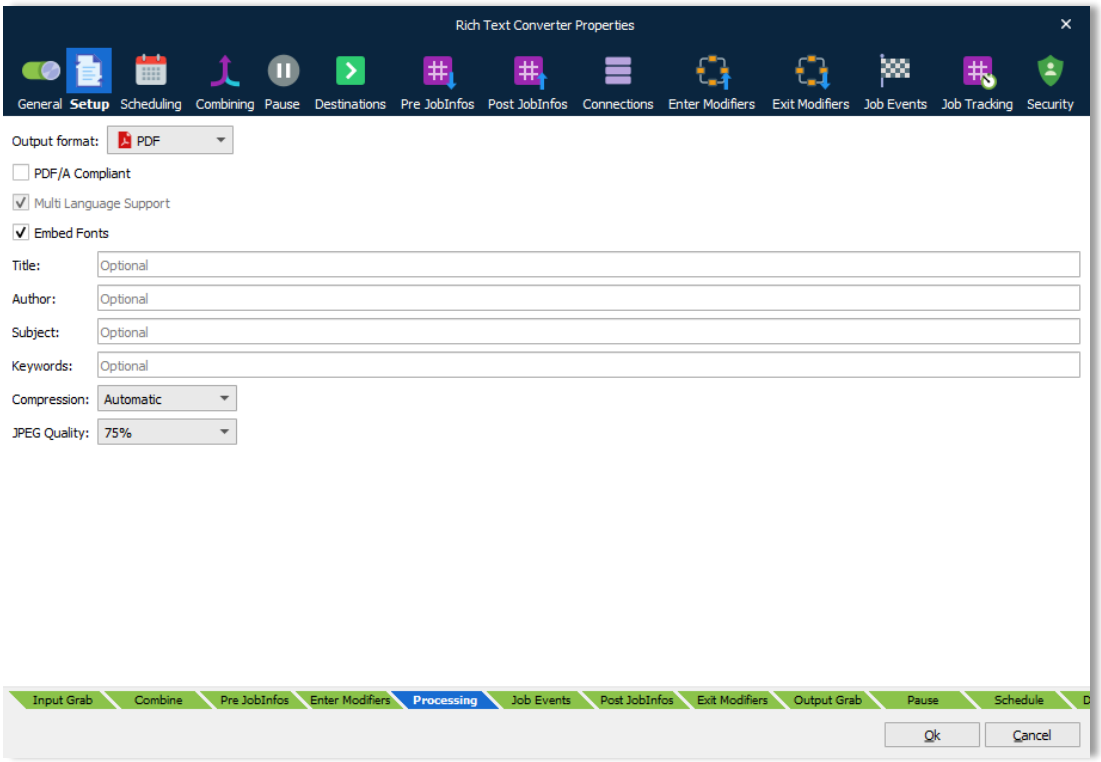
6.3.15 Process

The Process engine is used for running external processes. These processes can have data from Lasernet as input and produce output that is entered back into Lasernet. It is also possible to just run the external process and ignore any data transfer. For details on the Process engine see the section for the Process modifier.

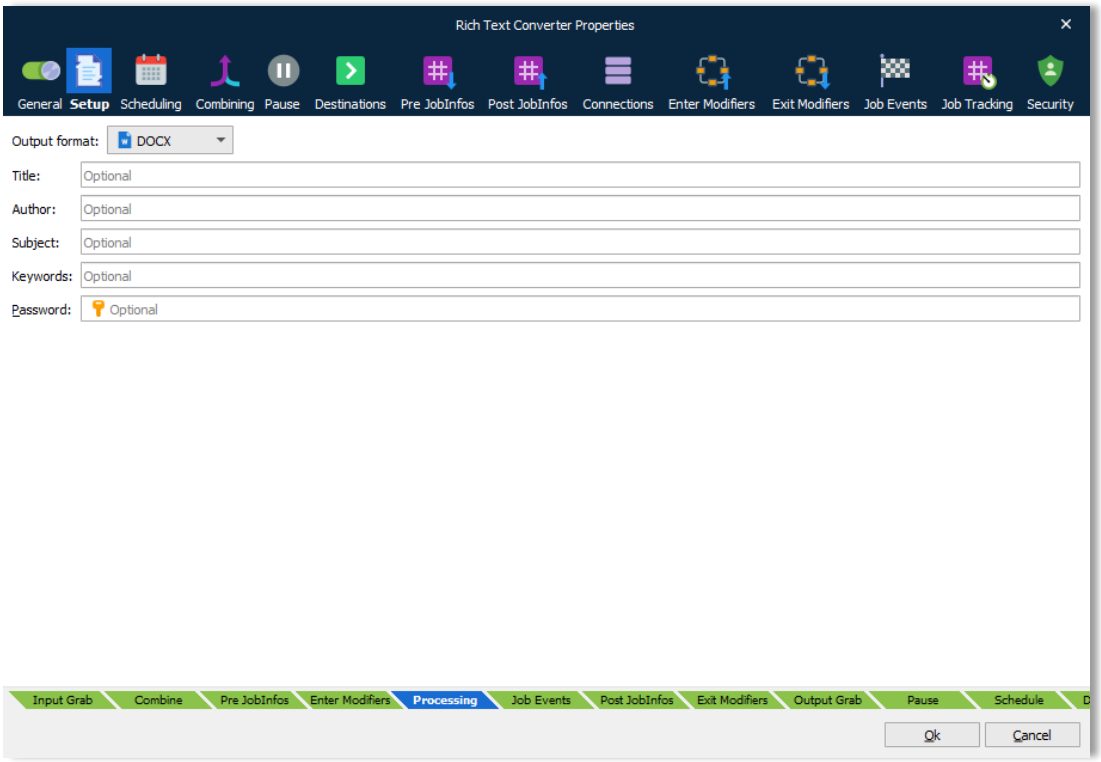


6.3.16 Rich Text Converter

Select PDF as the output format to convert a document, created in DOCX format, to a PDF or PDF/A compliant document.

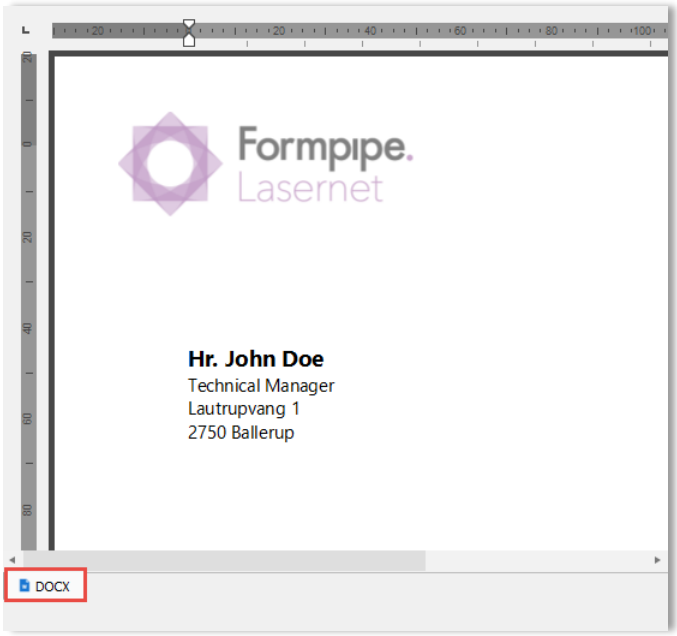


Select **DOCX** to add additional phrases to a DOCX document.



6.3.16.1 PDF

The Rich Text Converter Engine can only guarantee successful conversion of documents created in DOCX format by the Lasernet Form Editor.



Disclaimer: Lasernet cannot guarantee that all element types included in the DOCX format, created by Microsoft Word or other 3rd party applications, are successfully converted to PDF. The use of this module under these circumstances is considered experimental.

6.3.16.2 PDF/A Compliant
PDF/A is a PDF format typically used for the long-term archiving of electronic documents and is based on PDF reference version 1.4. Fonts and color profiles will be embedded in the PDF file.

6.3.16.3 Multi Language Support
By default, the PDF format includes 7-bit ASCII characters only. Multi language support allows embedding additional characters and fonts.

6.3.16.4 Embed Fonts
Determines what fonts are embedded in the PDF file. Font embedding is used to ensure correct output on other client computers.

6.3.16.5 Title, Author, Subject and keywords
Descriptions to be included in the document properties of the PDF-file.

6.3.16.6 Image Compression
Three options are available for compressing images, including the JPEG quality value (default 80%):

Automatic	Prefers indexed and uses JPEG if image contains more than 256 colors.
Indexed	Creates indexed images (using a palette) if image contains no more than 256 colors. Otherwise store as bitmap. The setting will give the best image quality but the file size may increase.
JPEG	Compresses all images using JPEG. The setting will compromise image quality for smaller file size.

6.3.16.7 DOCX
The **DOCX** format is used to add additional phrases, created in Lasernet Phrase Editor, to a DOCX document created by the Lasernet Form Editor. From Microsoft Word a DOCVARIABLE + the name of the phrase is required as a part of the document.

Disclaimer: Formpipe cannot guarantee that all element types included in the DOCX format, created by Microsoft Word or other 3rd party applications, are successfully converted to PDF. The use of this module under these circumstances is considered experimental.

6.3.16.8 Combining
You can merge multiple DOCX files into a single DOCX or PDF document by activating the combing feature in the module settings, but it is an experimental feature. Combining multiple documents that contain complex formatting like headers, footers, indexing, page numbering etc. may not always be merged as expected.

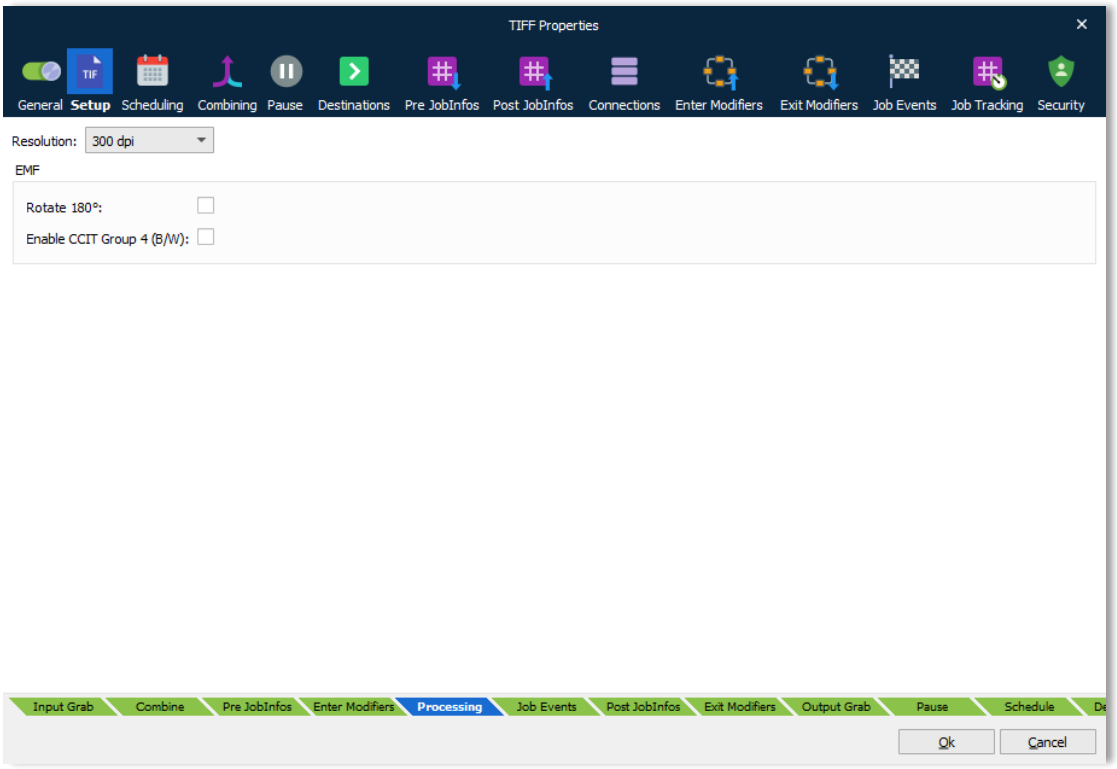
6.3.17 TIFF

TIFF module is used to convert EMF or PDF data to TIFF format and combine a mix of EMF and PDF into a single TIFF.

Paper size and orientation, in the incoming data, are automatically handled by the TIFF Modifier, by scanning the paper formats included in the EMF or PDF data.

6.3.17.1 Settings

The TIFF Engine has the following settings:



6.3.17.2 Resolution

The resolution can be set to 96 DPI, 192 DPI, 300 DPI or 600 DPI.

96 DPI or 192 DPI are used for most fax systems to define the fax quality.

300 DPI or 600 DPI are mostly used to store documents in archive systems.

6.3.17.3 Rotate

Activate 180 degrees if your fax system requires that the TIFF document is rotated before transmission.

6.3.17.4 Enable CCIT Group 4 (B/W)

CCITT Group 4 compression is a lossless method of image compression used in Group 4 fax machines. Used for black and white images only.

6.3.18 SOAP Web Service

Used in the same way as the SOAP Web Service Input module to retrieve or send data to a Web Service. See the section about the *SOAP Web Service* modifier for further information.

6.3.19 XLSX Merger

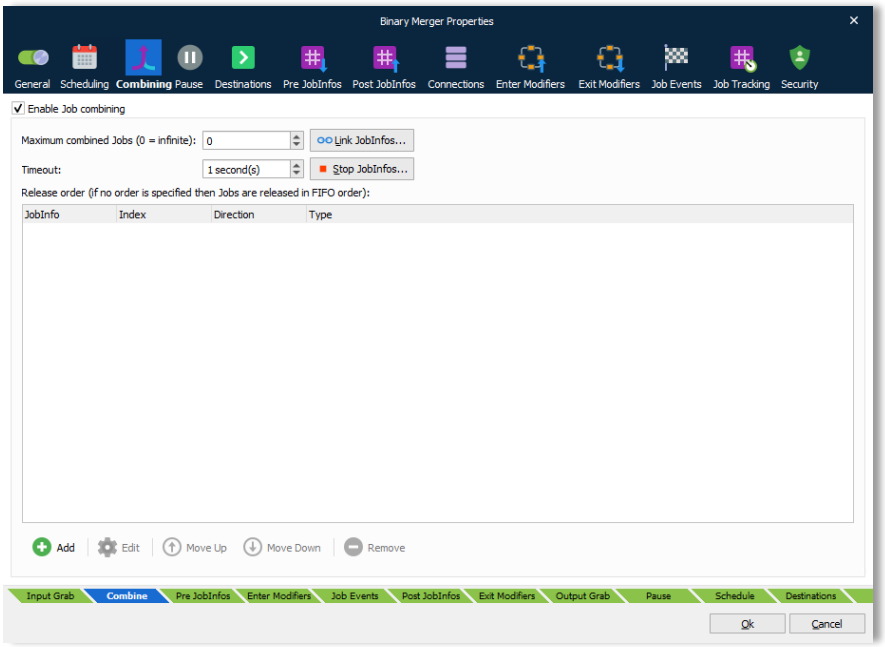
This engine combines a sheet from multiple spreadsheets, in the format XLSX, into a single spreadsheet containing multiple sheets.

The JobInfo named 'JobData' will contain the data for all the combined jobs. It will be attached to the JobInfos that were created in the first job. JobInfos for all other Jobs expect the first job will be deleted.

The XLSX Merger engine does not have any Setup tab.

6.3.19.1 Combining

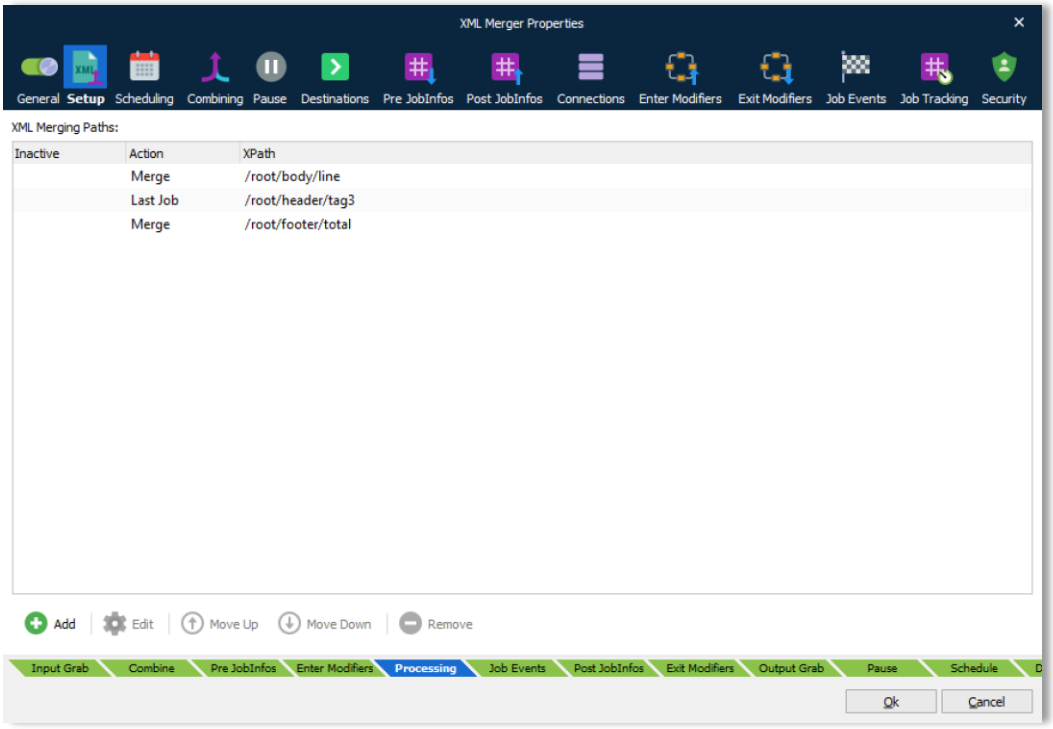
Select the **Combining** tab and activate **Enable Job combining** to merge sheets from multiple spreadsheets into a single spreadsheet. More information about combiner settings can be found in chapter 12.1.



6.3.20 XML Merger

The XML Merger engine can merge two or more similar XML documents into one.

The JobInfo named ‘JobData’ will contain the data for all the combined jobs. It will be attached to the JobInfos that were created in the first job. JobInfos for all other Jobs expect the first job will be deleted.



The first XML is the template for the output XML. It contains all data as well as the data merged from the other XMLs or inserted from the last XML.

The Combining rules determine which XMLs are merged.

6.3.20.1 Actions:

Merge Taken from all documents in the batch.

Last Job Taken from the last document in the batch.

In this example, we have an output structure, which will use the first XML, plus all body/lines from additional XMLs and finally the header/tag3 and footer/total from the Last XML.

JobData in job no. 1:

```
<root>
    <header>
        <tag1>1</tag1>
        <tag2>1</tag2>
        <tag3>1</tag3>
    </header>
    <body>
```

```
        <line>1</line>
    </body>
    <footer>
        <total>1</total>
    </footer>
</root>
```

JobData in job no. 2:

```
<root>
    <header>
        <tag1>2</tag1>
        <tag2>2</tag2>
        <tag3>2</tag3>
    </header>
    <body>
        <line>2</line>
    </body>
    <footer>
        <total>2</total>
    </footer>
</root>
```

JobData in job no. 3:

```
<root>
    <header>
        <tag1>3</tag1>
        <tag2>3</tag2>
        <tag3>3</tag3>
    </header>
    <body>
        <line>3</line>
    </body>
    <footer>
        <total>3</total>
    </footer>
</root>
```

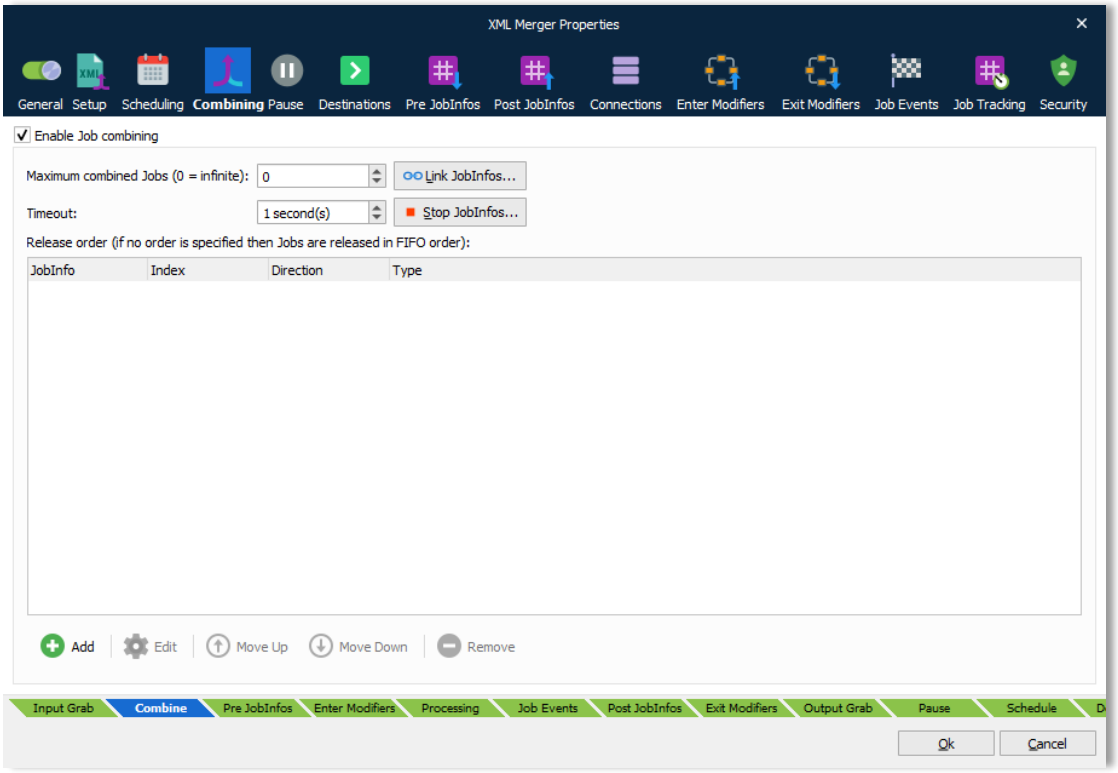
Our resulting XML will look like this (without the comments):

```
<root>
    <header>
        <tag1>1</tag1>    //generated from JobData1
        <tag2>1</tag2>    //generated from JobData1
        <tag3>3</tag3>    //generated from JobData3
    </header>
    <body>
        <line>1</line>    //generated from JobData1
        <line>2</line>    //generated from JobData2
        <line>3</line>    //generated from JobData3
    </body>
```

```
    <footer>
        <total>1>/total>  //generated from JobData1
        <total>2>/total>  //generated from JobData2
        <total>3</total>  //generated from JobData3
    </footer>
</root>
```

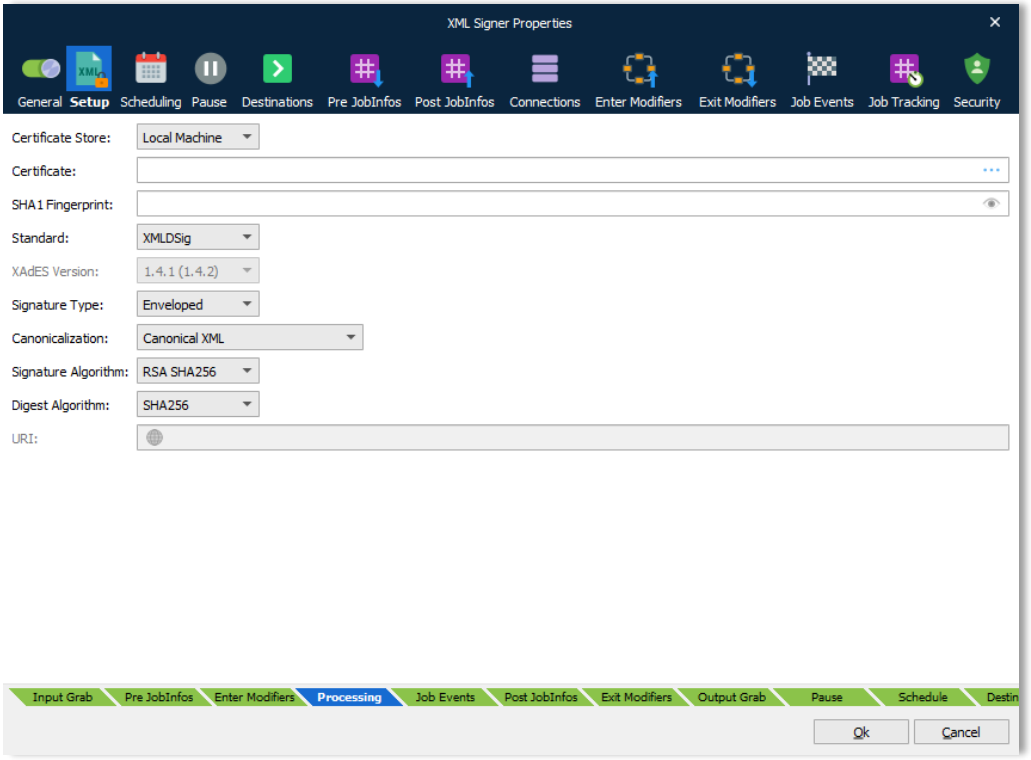
6.3.20.2 Combining

The **Enable Job combining** box must be activated on the Combining tab (in the XML Merger module), to make use of the action rules defining how the XML data will be merged. More information about combiner settings can be found in chapter 12.1.



6.3.21 XML Signer

By signing a XML file using a digital certificate, you allow the recipient to verify both the origin and the integrity of the file. This means that the recipient is able to verify that the XML is the original and has not been changed after it was created.



6.3.21.1 Certificate Store

Select whether the certificate is located on the Local Machine or the Current User account.

6.3.21.2 Certificate

In order to sign the PDF using a digital signature, you must have a valid and appropriate certificate installed for the user account that the Lasernet service is running under.

6.3.21.3 SHA1 Fingerprint

Used to verify that the signed file is unaltered. The checksum is created before the file is transmitted, and then once again when it reaches its destination.

6.3.21.4 Standard

XML Digital Signatures (XMLDSIG) or XML Advanced Electronic Signature (XAdES) are supported for signing and validating.

6.3.21.5 XAdES Version

Set version for XML Advanced Electronic Signature (XAdES).

- 1.2.2
- 1.3.2
- 1.4.1 (1.4.2)

6.3.21.6 Signature Type

- Enveloping signature is over content found within an object element of the signature itself. The object (or its content) is identified via a reference (via a URI fragment identifier or transform).
- Enveloped signature is over the XML content that contains the signature as an element. The content provides the root XML document element.
- Detached signature must be placed in a separate file next to the XML document

Note: The signature data is placed in JobData and overwrites the original XML contents. XML data must be stored in a separate JobInfo or forwarded to an additional destination before signing.

6.3.21.7 Canonicalization

Canonical XML specifies a standard serialization of XML that, when applied to a sub document, includes the subdocument's ancestor context including all of the namespace declarations and attributes in the "xml:" namespace:

- Canonical XML
- Canonical XML with comments
- Canonical XML 1.1
- Canonical XML 1.1 with comments
- Exclusive
- Exclusive with comments
- Minimal

6.3.21.8 Signature Algorithms

Specifies the hash to protect from tampering:

- DSS
- RSA MD5
- RSA SHA1
- RSA SHA224
- RSA SHA256
- RSA SHA384
- RSA SHA512
- ECDSA SHA1
- ECDSA SHA224
- ECDSA SHA256
- ECDSA SHA384
- ECDSA SHA512

6.3.21.9 Digest Algorithms

Specifies the hash algorithm before applying the hash.

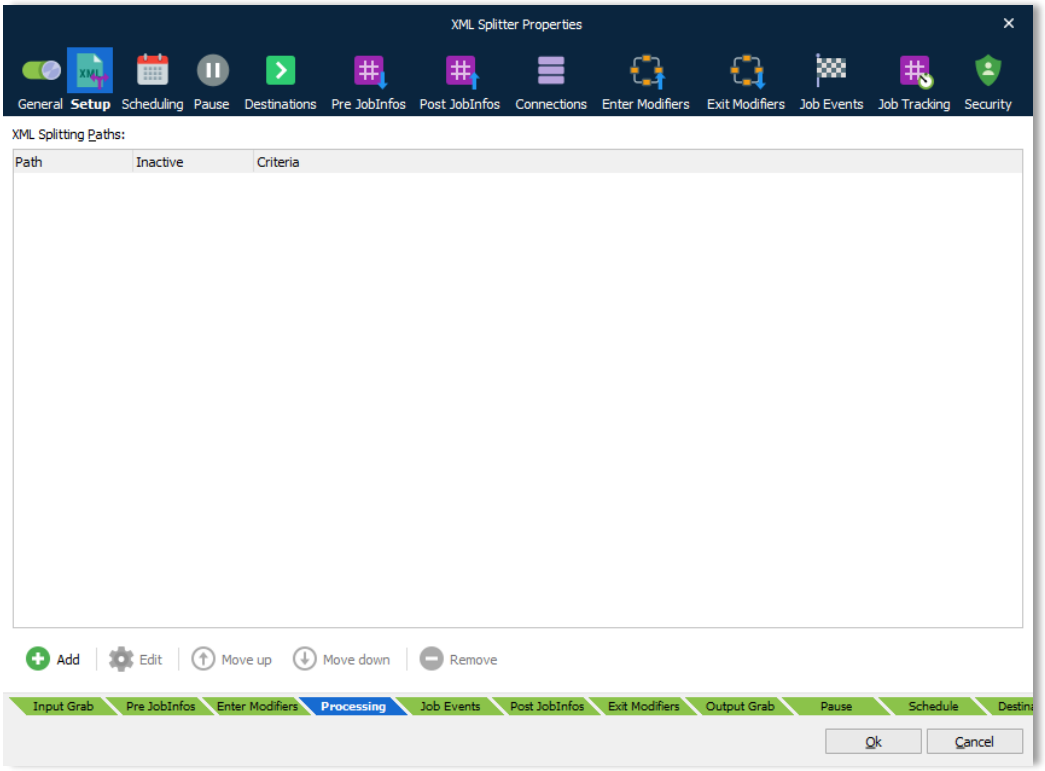
- MD5
- SHA1
- SHA224
- SHA256

6.3.21.10 URI

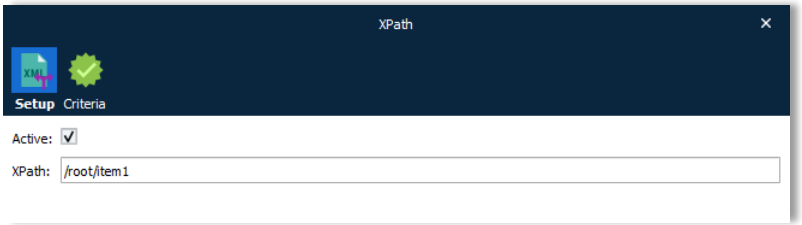
Sets the Uniform Resource Identifier.

6.3.22 XML Splitter

The XML Splitter module can split XML files into smaller fragments to ease the load on the Form engine.



The XML Splitter module will split the XML file on each of the paths listed in the *XML Splitting Path* and create a new job for each. Any XML not mentioned in the *XML Splitting Path* and not a child of any of those will be present in all jobs.



For each XPath, you can attach a JobInfo Criteria. The split action will only run if both the XPath and criteria are true. If the list of XML Splitting Paths contains multiple XPaths, any XPath that is true will run for the very same XML document.

The table illustrates how two sets of split paths separate a document into three and four jobs, respectively.

Split paths / Original document	root/item1	root/item1 root/item2
<root> <header/> <item1/> <item2/> <item1/> <item1/> <footer/> </root>	<root> <header/> <item1/> <item2/> <footer/> <root> <header/> <item2/> <item1/> <footer/> <root> <header/> <item2/> <item1/> <footer/>	<root> <header/> <item1/> <footer/> <root> <header/> <item2/> <footer/> <root> <header/> <item1/> <footer/> <root> <header/> <item1/> <footer/>

6.3.22.1 JobInfos

The XML Splitter sets some JobInfos when creating a Job.

FirstFragmentInXMLJob Set to 1 (true) when first fragment of the XML is created.

LastFragmentInXMLJob Set to 1 (true) when last fragment of the XML is created.

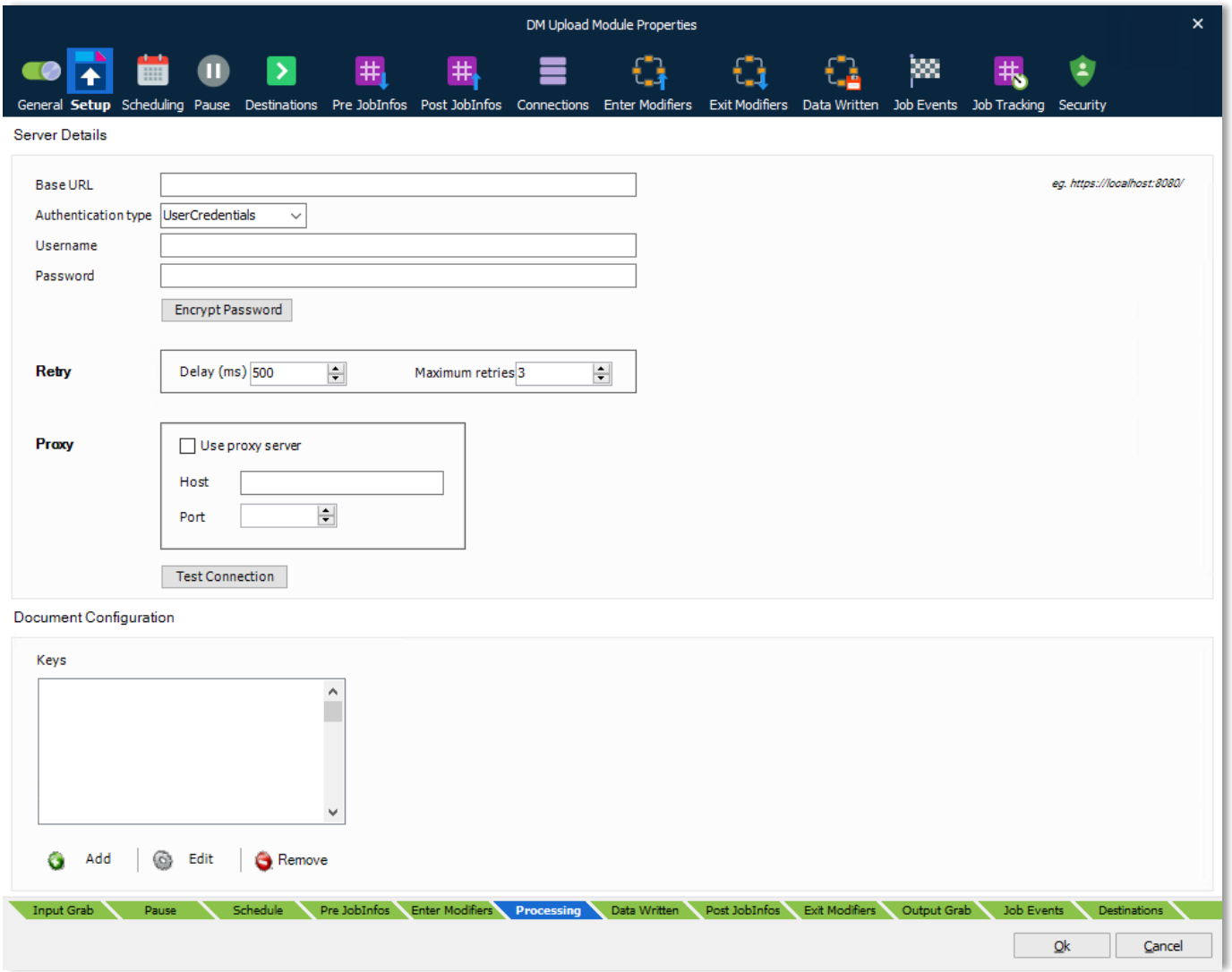
6.4 Output Modules

6.4.1 Autoform DM Upload

The module is designed to provide a simple yet effective method for archiving documents and associated metadata to Autoform DM. As such the configuration process is relatively straightforward. There are two different ways that you can use the Autoform DM Upload Module. The first, and most common way, is as an output module, to archive a finished (processed) document. The second way, is as an engine to retrieve the DocumentID and DocumentRevision.

For more information on migrating your forms and builds from using JFinder to the Autoform DM Upload Module, please see the separate Migration Guide document which is available upon request. Users creating new forms from scratch may also find this document useful as it runs through some basic set up and configuration steps for archiving to Autoform DM.

6.4.1.1 Configuration



6.4.1.2 Server Configuration
Authentication Type

Note: Some dialog fields will change depending on the selected option.

There are three authentication options to choose from:

UserCredentials The default option. Authenticate requests using a username and password.

OAuth For use when Autoform DM is configured for SSO, use a token obtained via a modifier to authenticate the request – requires no direct configuration in the module. See OAuth section for more detail. Note: Requires Autoform DM 10.2 or later.

Apikey Authenticate requests using an API Key generated by Autoform DM; this should be the preferred option unless using SSO. Note: Requires Autoform DM 10.0 or later.

Depending on authentication scheme selected, the following configuration is available:

Scheme	Option	Description
All	Base URL	URL Of Autoform DM Server. Can either be specified using servername, IP Address or environment variable (for more information see Environment Variables section)
All	Authentication type	Select an option from the drop-down menu.
UserCredentials	Username	User with access to Autoform DM Server
UserCredentials	Password	Password for user
UserCredentials	Encrypt Password	Encrypt the provided Autoform DM server password in the environment (for more information see Encryption section)
Apikey	API Key	Enter the API Key generated by Autoform DM
Apikey	Encrypt API Key	Encrypt the provided Autoform DM API Key in the environment (for more information see Encryption section)

Proxy Configuration (if required): For proxy configuration details, please consult your IT systems administrator.

Use Proxy Server Select the checkbox if you use a proxy server to connect to Autoform DM

Host IP or Hostname of your proxy server

Port Port number to connect to your proxy server

6.4.1.3 Environment Variables
If you have a number of different environments (such as SIT, UAT & Live) which all archive to different Autoform DM Servers, you can use environment variables (ENV) instead of fixed values for the Base URL, Username and Password, in order to avoid having to change these values every time a build is moved between environments. To make use of this feature, use the following special values in the Server Configuration panel to point to your systems' environment variables table.

Base URL `ENV[DM_BASEURL]`

Username `ENV[DM_USERNAME]`

Password `ENV[DM_PASSWORD]`

Ensure that your OS' System Environment Variables contains the corresponding entries from which the Autoform DM Module can extract these values. They should be added in the following format:

```
DM_BASEURL=
DM_USERNAME=
DM_PASSWORD=
```

e.g.

```
DM_BASEURL=http://testserver
DM_USERNAME=admin
DM_PASSWORD=thisisapassword
```

6.4.1.4 OAuth

This section applies if you selected OAuth as the Authentication type – this is only applicable when Autoform DM is configured for SSO (Single Sign-On) and authentication is performed against an external system rather than Autoform DM directly.

When selected, no further configuration related to authentication is done in the module, instead the Lasernet OAuth2 modifier is used to obtain the required token which the upload module will then use for requests.

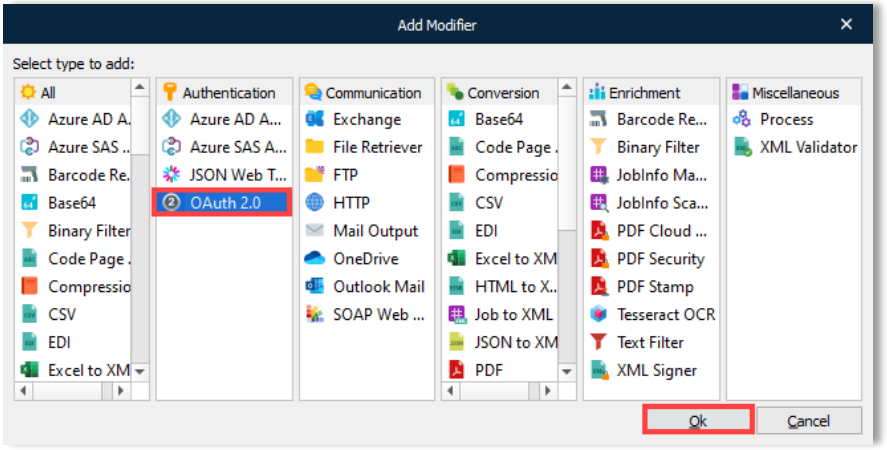
Failure to correctly add the modifier will result in the Upload Module showing an error indicating this.

Also note that the ability to 'test the connection' will fail as this is not valid when this option is selected.

Follow the steps to enable the OAuth modifier – in order to complete this you will need a **Client ID**, a **Client Secret**, and a **Token endpoint URL** – these will be available based on the SSO configuration:

1. Select the Enter Modifiers tab on the Upload Module configuration
2. Click **Add**
3. Click **Add…** in the Choose modifier dialog

4. Select OAuth 2.0 in the Add Modifier dialog Authentication column, then click **Ok**

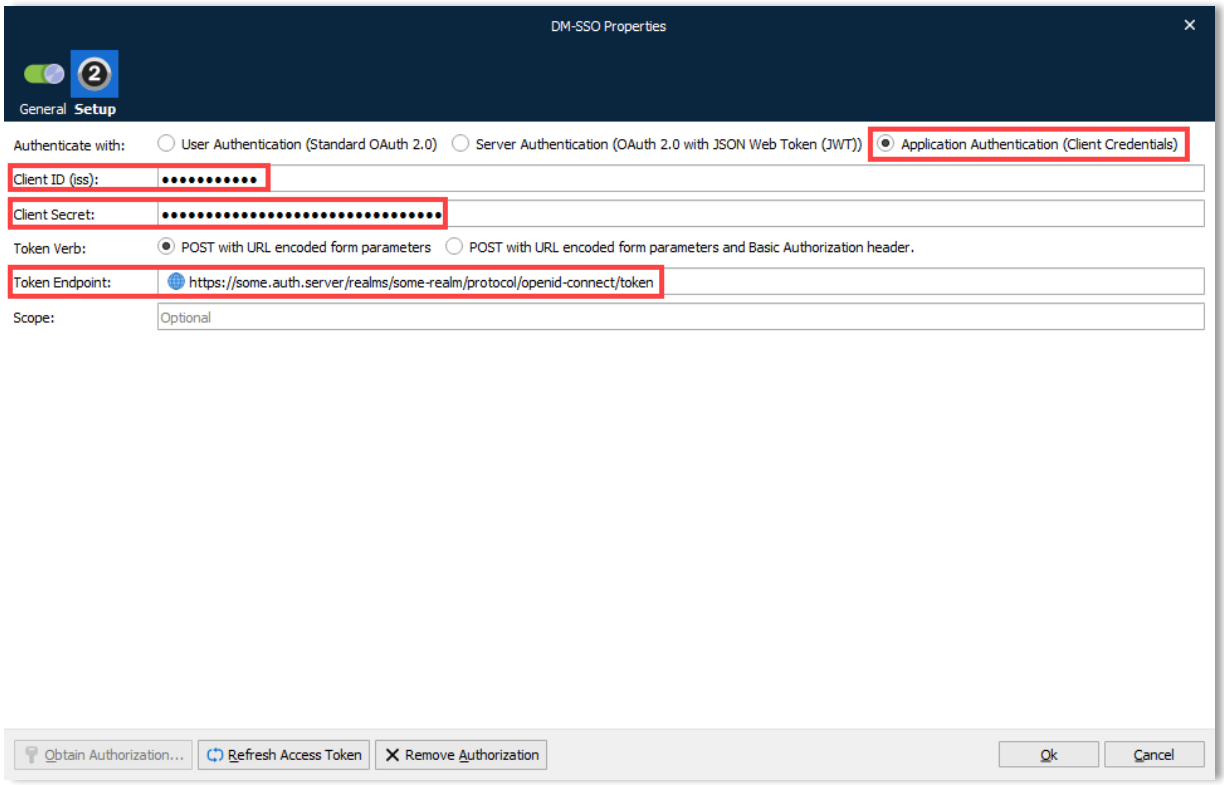


5. Select the Setup tab in the Modifier Properties tab
6. Select the Application (Client Credentials) radio button
7. Enter the Client identifier into the Client ID (iss) field. For example,

```
autoform-dm
```

8. Type the Client secret into the Client Secret field
9. Type the endpoint URL into the Token Endpoint field. For example,

```
https://some.auth.server/realms/some-realm/protocol/openid-connect/token
```

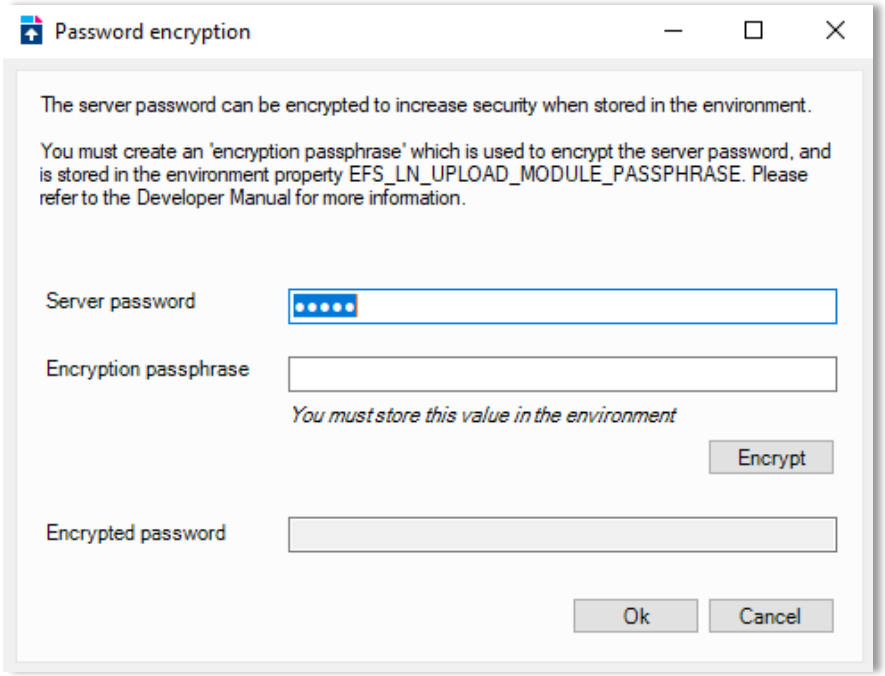


10. Click **Ok**

To verify the entered configuration is correct, click the **Refresh Access Token** button.

Note: To edit the properties, double-click the modifier, then click **Edit…** in the Choose modifier dialog.

6.4.1.5 Encryption



Whilst Lasernet stores the Autoform DM credentials in an encrypted form within the build, if you are using the Environment Variables functionality, we recommend that you encrypt the DM password before storing it in the environment. To encrypt the password within the module, follow the instructions below:

- Make sure the password field in the server setup window, has the correct server password (not an environment reference)
- Click the 'Encrypt Password' button, which will launch the Password Encryption dialog
- Add an Encryption Passphrase to encrypt the server password
- Follow the instructions on the screen and store the passphrase in the environment
- Click Ok to encrypt the value and close the dialog
- Now reveal the password field - you can see that it is encrypted. You can now put that value in the DM_PASSWORD environment variable and replace the value in the server password field with ENV[DM_PASSWORD]

Note: When adding the encrypted password to the environment, ensure that it is enclosed with `ENC[ … ]` (see example below) to signify that it has been (ENC)rypted.

```
DM_PASSWORD=ENC[]
EFS_LN_UPLOAD_MODULE_PASSPHRASE=
```

e.g.

```
DM_PASSWORD=ENC[rL+pR2yVTfUwXczqMwG10g==]
EFS_LN_UPLOAD_MODULE_PASSPHRASE=ln_test_passphrase
```

6.4.1.6 Document Configuration

The keys provided in the document configuration box represent the jobinfos on the form that will be picked up by the module and used for archiving. The module will also check the document definition in Autoform DM to ensure the keys are valid (bound) for that particular definition. If the key does not exist or is not bound, the upload will fail.

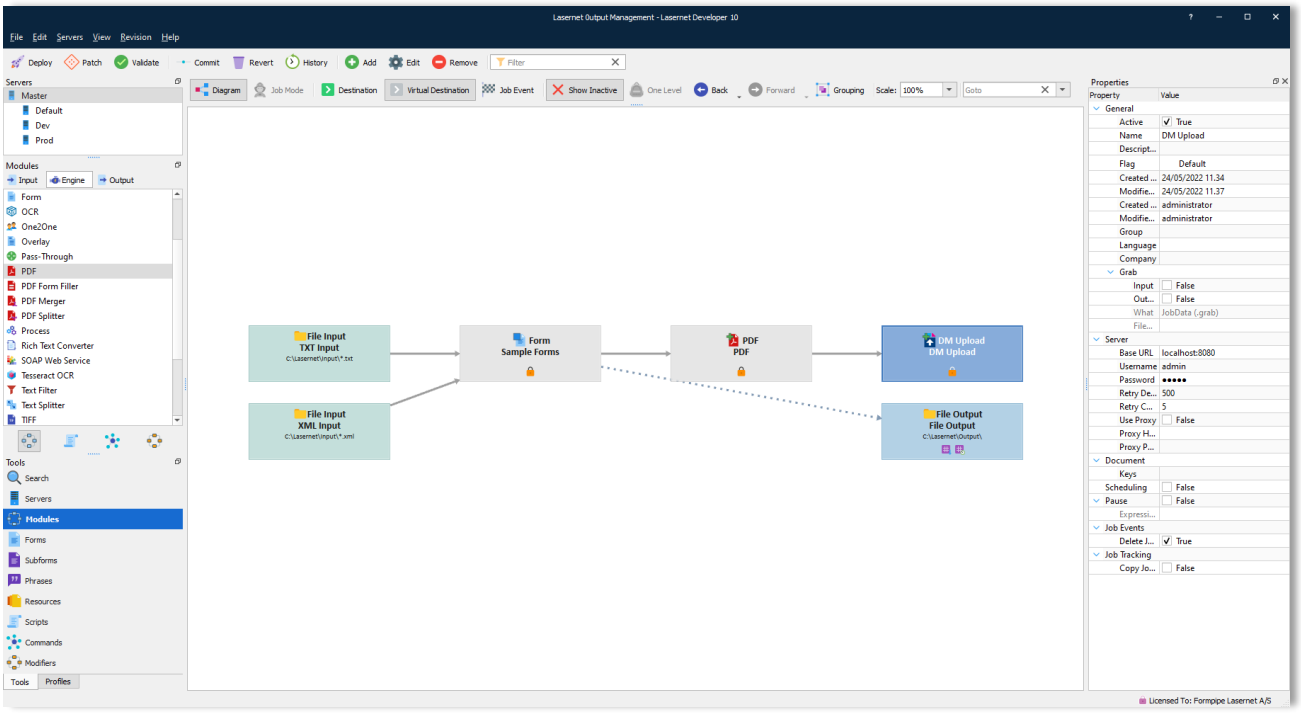
For the Autoform DM Upload module to run successfully the required keys must:

- Be defined on the form
- Be listed in Document Configuration area in the Autoform DM Upload Module
- Be configured in Autoform DM and bound to the correct Document Definition.

6.4.1.7 Connect

To make use of the newly configured Autoform DM Upload module, set it as the destination from the appropriate processing engine.

The example below shows the AUTOFORM Upload Module as an output destination for the PDF Engine.



6.4.2 Advanced Processes

6.4.2.1 Autoform DM Upload JobInfos

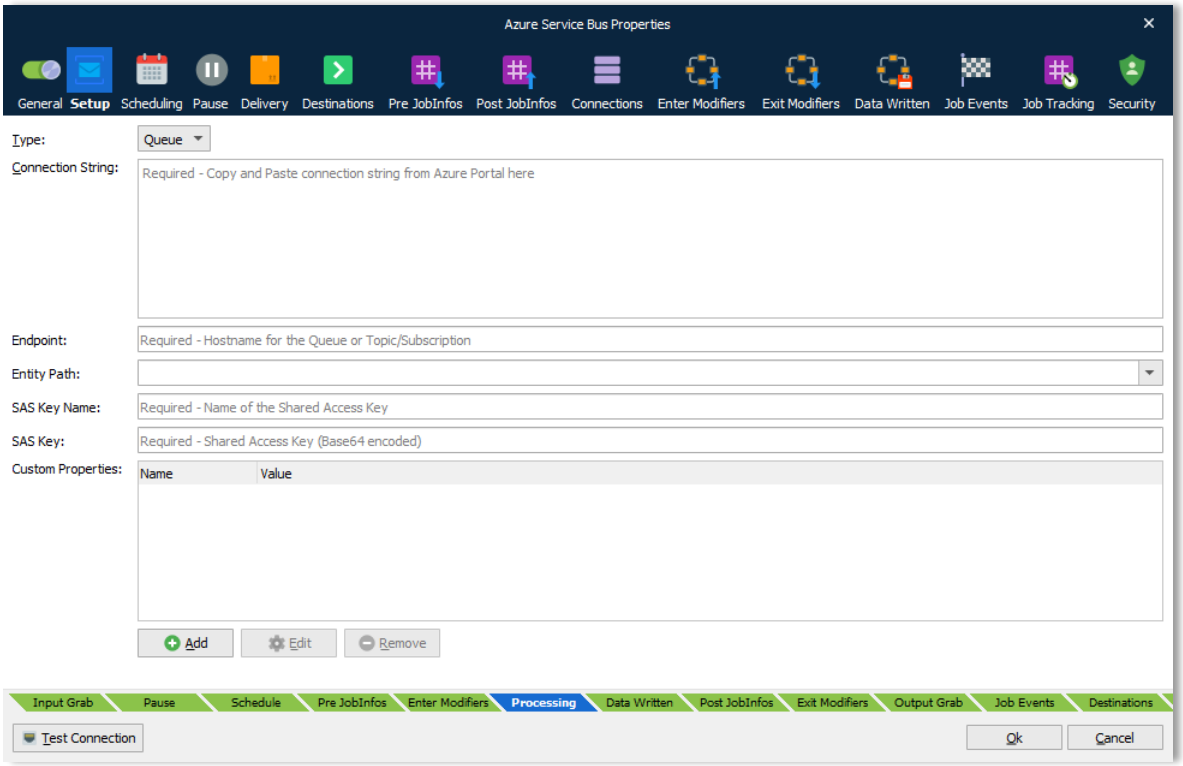
The following new JobInfos have been created to give greater control over the Autoform DM Upload functionality.

JobInfo	Example JobInfo value	Default value	Description
DocumentDefinition	account-statement	(None)	Document definition name in Autoform DM
DocumentDateFormat	yyyy-MM-dd	System locale	Date format for the metadata keys to be archived with this document. This is the same as JFinder's PDMDFMT
DocumentUpdate	true	false	Whether to update a document based on matching Autoform DM ID or CUK (Customer Unique Key). Where set 'false' and an ID or CUK jobinfo has been used, the job will fail if a matching document already exists in the Autoform DM archive. This is the same as JFinder's PDMUPD. Note: The DocumentCUK or the DocumentID need to be specified when setting DocumentUpdate to 'true'.

JobInfo	Example JobInfo value	Default value	Description
DocumentAttach	false	true	This controls whether the document is archived, or just the metadata. This is the same as JFinder's PDMATT
DocumentID	8458941655	(Empty)	Autoform DM Document ID for an existing document to update
DocumentCUK	customer-123-stmt-2018-06	(Empty)	Autoform DM Document CUK (Customer Unique Key) for an existing document to update

6.4.3 Azure Service Bus

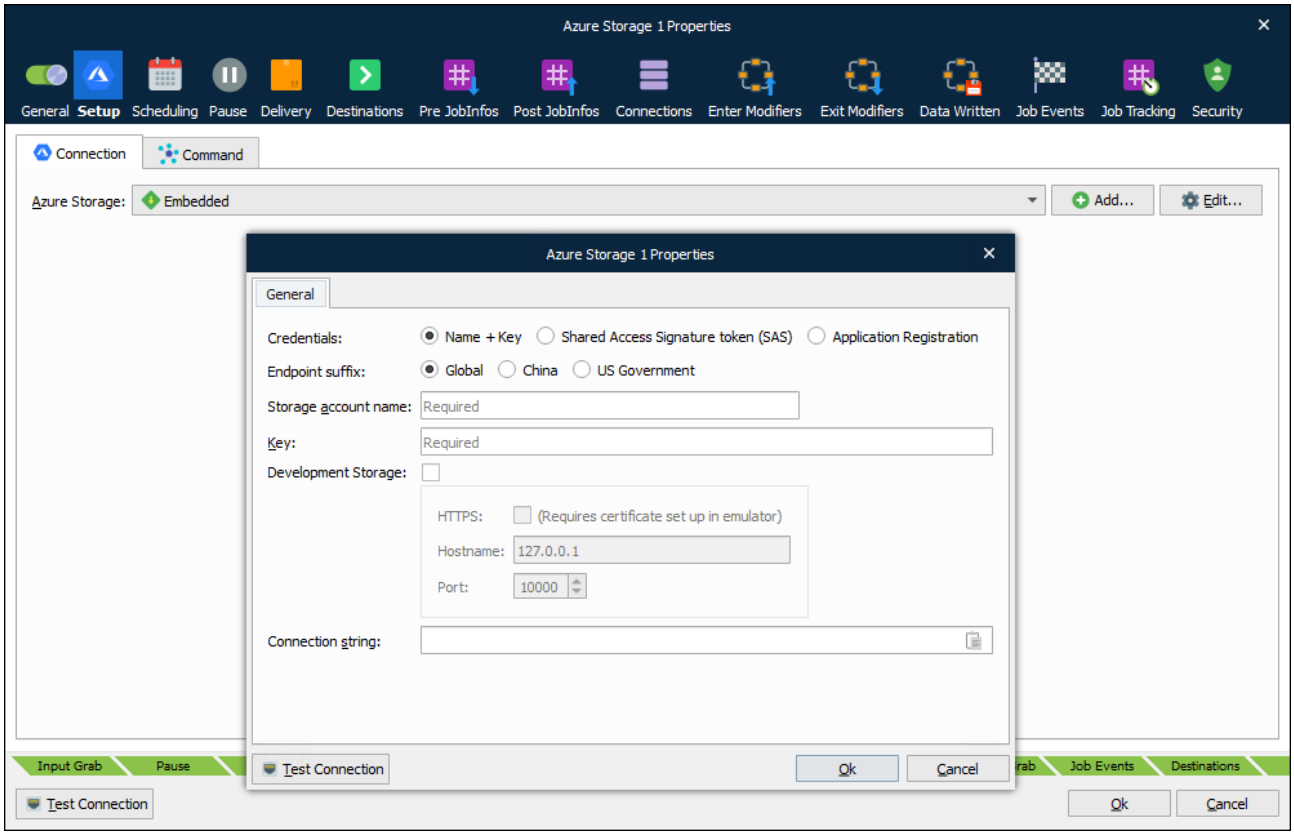
Used for connecting to the Azure Service Bus.



Please refer to the “Lasernet 10 – Azure” manual for more information.

6.4.4 Azure Storage

Used for connecting to Azure Storage.



Refer to the “Lasernet 10 – Azure” manual for more information.

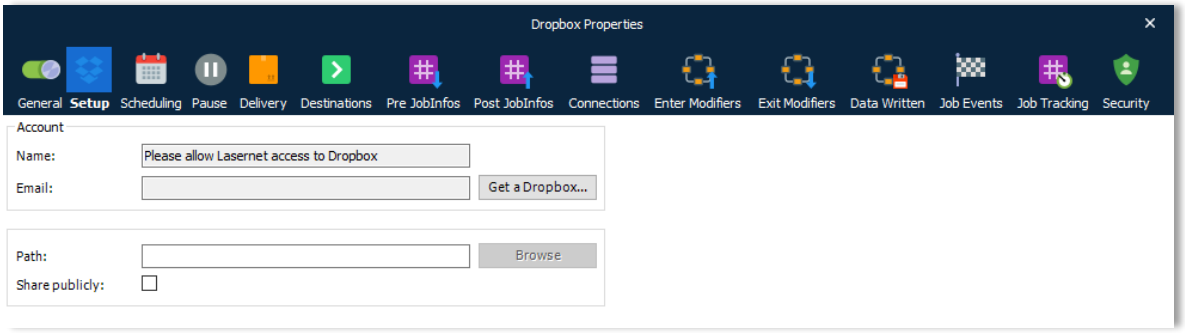
6.4.5 Dropbox

Dropbox is a service that lets you bring all your documents in to the cloud.

Important: This module is deprecated in Lasernet and will not be supported in future versions of Lasernet.

6.4.5.1 Output

Lasernet can save files in a Dropbox folder.



JobInfo substitution is supported for the JobInfo 'Filename'. An ID is used in the Dropbox backend instead of the path shown in the module, therefore the Path cannot be substituted.

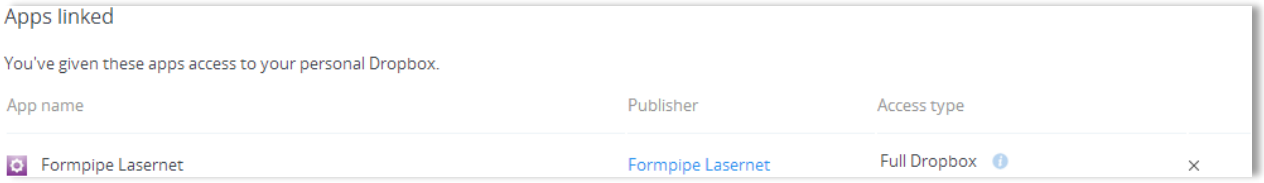
Lasernet will always overwrite files in the same folder with the same name.

6.4.5.2 Authentication

Dropbox uses OAuth 2.0 authentication. The OAuth 2.0 authorization framework enables Lasernet to obtain limited access to Dropbox by orchestrating an approval interaction between the resource owner and Dropbox. This basically means that you need to give Lasernet access to your Dropbox.

Giving Lasernet access to a Dropbox is done by pressing the "Allow Lasernet" button, which launches a browser window where the user can log into the Dropbox account and allow Lasernet access to it. This action stores an encrypted token in the configuration. This token is used to regain access to the Dropbox at a later stage.

Revoking access is done via the web interface for Dropbox where it's possible to unlink an app, in this case Formpipe Lasernet, from Dropbox.

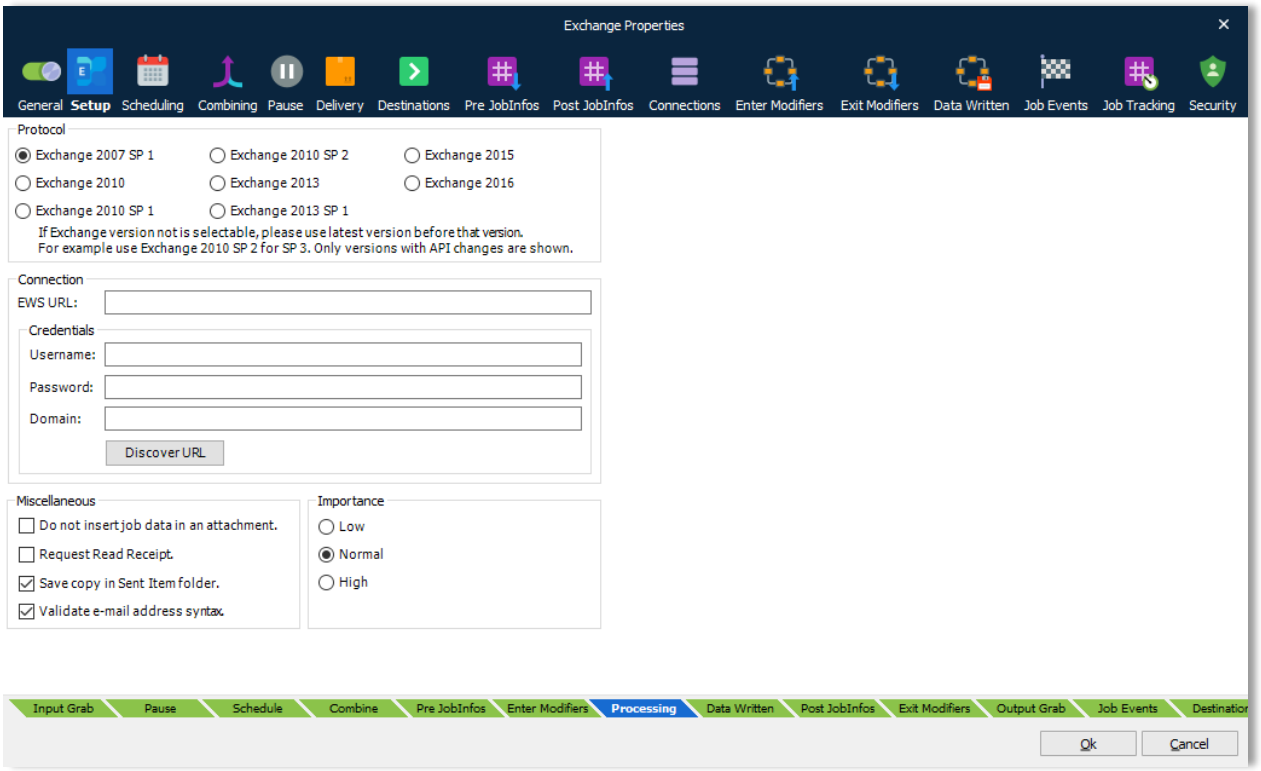


6.4.5.3 JobInfos

DropboxURL	URL to the publicly shared file. The JobInfo is only set when **Sharing publicly** is enabled in the GUI.

6.4.6 Exchange

Used for sending emails via Microsoft Exchange Server. The Exchange module integrates directly with Exchange Web Services (EWS) to send emails.



6.4.6.1 General

Protocol

Version of the Exchange server. If your Exchange version is not selectable, please use the latest version in the list. As an example, you can use Exchange 2010 SP2 for SP3. Only versions with API changes are selectable.

EWS URL

Location of the Exchange Web Services. If Microsoft Online (Cloud solution) is used, the following URL can be used:

https://red002.mail.emea.microsoftonline.com/ews/exchange.asmx

or

https://amsprd0310.outlook.com/EWS/Exchange.asmx

If the URL is left empty, Lasernet will run an auto discover service each time the poll interval is reached. If the URL is retrieved successfully, it will be logged and a connection will be established. We recommend that you add the URL stored in the log, to the EWS URL field to enable faster connection to the Exchange Server in future.

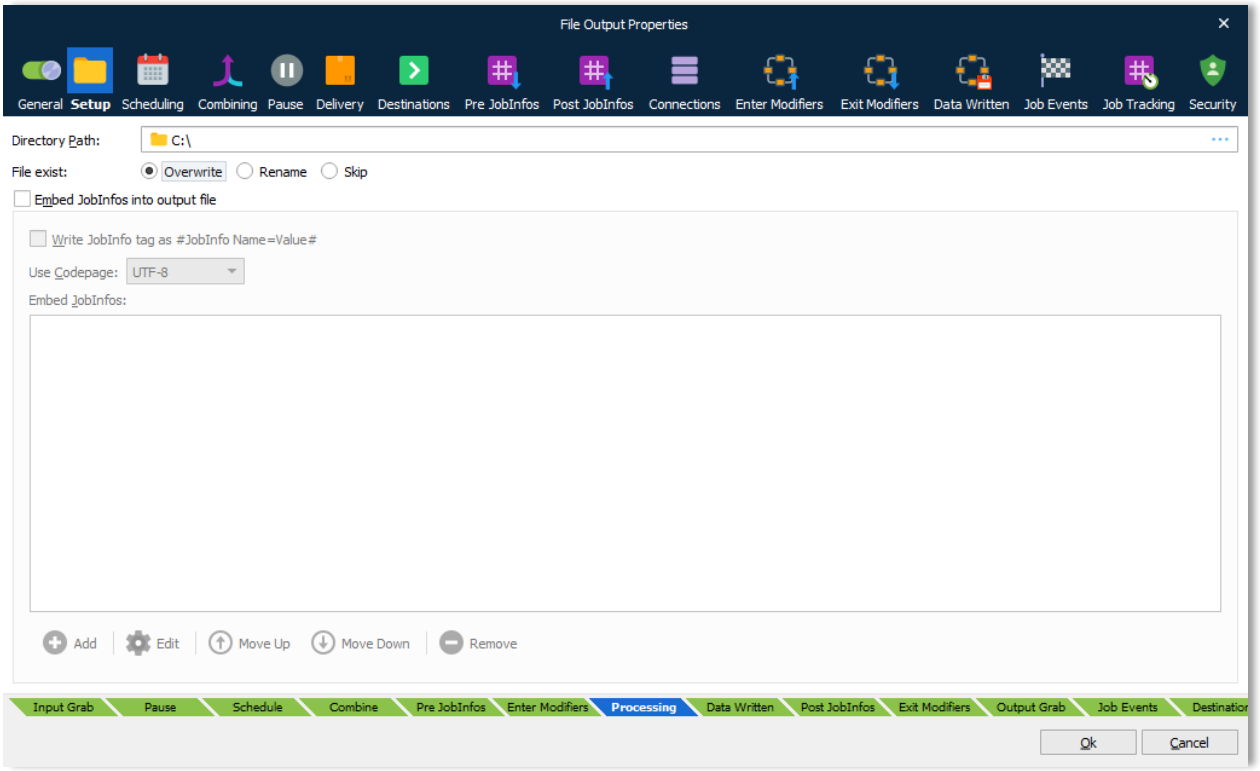
Username	Email address or name of active directory user
Password	Password for the account
Domain	A domain name is required for active directory users
Do not insert job data in an attachment	Activate this setting to send an email without including JobData as an attachment. EWS server will as report a non-critical error log message when sending emails without attachments: Unable to read MimeContent; This operation can't be performed because this service object doesn't have an Id.
Request Read Receipt	Activate if you want a receipt that the addressee has opened the mail.
Save copy in Sent Item folder	Activate if you want to store a copy of the mail in the Sent Item folder in your email client.
Validate e-mail address syntax	Activate to validate the email syntax in MailTo, MailCC and MailBCC before sending the email to the mail server. If the syntax is not valid, the job will fail. To prevent a job from failing the setting must be de-activated. This will allow mail addresses to contain syntax used for external fax systems connected to mail servers. Comma and semicolon are valid as separators if MailTo, MailCC or MailBCC contain multiple addresses.
Importance	Set the category of importance to Low, Normal (default) or High.

EWS was introduced by Microsoft in Exchange Server 2007, but will no longer receive feature updates. While the service will continue to receive security updates and certain non-security updates, product design and features will remain unchanged.

Microsoft has announced that EWS will still be available and supported for use in production environments. However, we suggest migrating to the Outlook Mail module, using Microsoft Graph API to access Exchange Online data and gain access to the latest features and functionality. The Outlook Mail module was introduced in Lasernet 9.5

6.4.7 File Output

Used for sending jobs to a directory.



Directory Path

Sets the directory or UNC path where to store files. By default, the maximum length for a path is 260 characters. The filename should not be included.

A local path is structured as follows: C:\Lasernet\Output\ or \\<Local machine>\<Local path>\

An extended length path is supported for a maximum total path length of 32,767 characters. To specify an extended length path you can use a prefix: \\?\C:\<Very long local path>\ or \\?\<Very long local path>\

To support an extended length path for a shared network drive type: \\?\UNC\<Server>\<Path>\

The directory path can also be stored in a JobInfo. A JobInfo substitution is defined as the value of the directory path, e.g.: #UNCDirectoryPath#

File exist

Behaviour for files that already exist. By default, the file will be overwritten. If rename is activated, a splitter symbol and number of digits for suffix can be defined. Skip setting is used to skip file or fail a job.

When printing to a File Output module the “Filename” JobInfo attribute sets the filename of the file produced. The file is created in the directory path entered into the Directory Setup dialog.

6.4.7.1 Embed JobInfos into output file

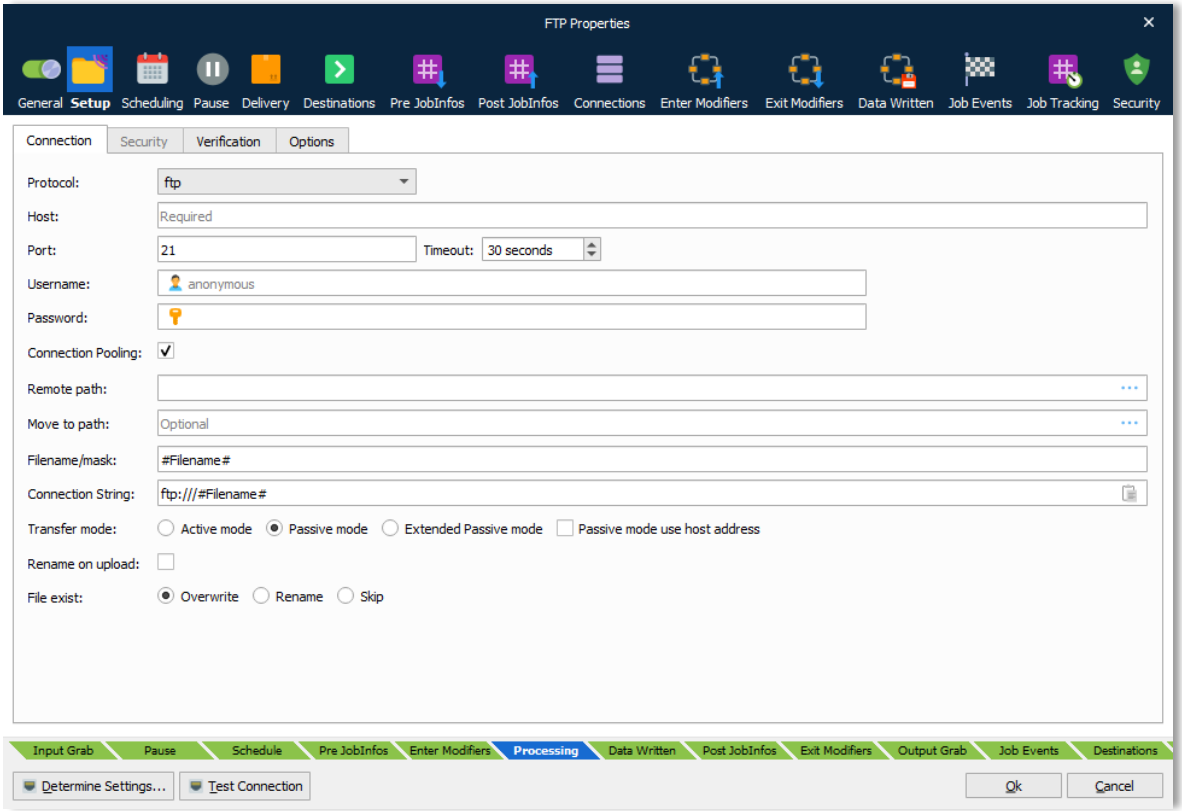
It is possible to make the File Output module insert specific JobInfos into the file. This is useful if the file is to be picked up by Lasernet again (the File Input module can collect the embedded JobInfos automatically) or if other programs have support for extracting the data. To enable insertion of JobInfos, check the Embed JobInfos checkbox. The JobInfos will then be placed at the beginning of the file before the contents of JobData.

The JobInfo keywords can be embedded as “#JobInfo Name=Value#”.

6.4.8 FTP

Used for sending jobs to an Internet URL destination such as an FTP, FTPS or SFTP server.

To connect to an FTP server, select the protocol and enter the address of the server into the host field. Add the server port number into the port field (defaults to 21 unless specified). If a username and password is required, enter it in the corresponding fields (defaults to anonymous logon if no credentials are specified). Click **Determine Settings** to validate against the specified host and retrieve information for supported ports, transfer modes, protocols, SSL types and Clear Control Channel (active/not active).



6.4.8.1 Connection

Protocol	Supported protocols are FTP, FTPS and SFTP
Host	URL for FTP. Protocol type or remote path should not be included as part of the URL.
Port	Set port number used by host. Default for FTP is port 21 and for SFTP port 22.
Timeout	If the connection cannot be established before the timeout limit it will fail. Default timeout is 30 seconds.
Username and password	If user authentication is required, enter a username and password, otherwise anonymous logon will be used by default.

Connection Pooling

Connection pooling is activated by default when adding a new FTP module. It works as a cache to reuse the open connection when other requests to the same FTP server are required. Connection pools enhance performance, since opening and closing an FTP connection is costly and wastes resources.

Important: The combination of using secure FTP and the Windows Firewall can cause problems when pooling connections. The symptoms are either missing or 0-byte files on the server and a failed Job in Lasernet stating that connection was forcibly closed. What actually closes the connection is the Windows Firewall.

The solution is either to not use pooling, which seriously slows down processing if many files are uploaded to the same server, or to disable stateless filtering in the Windows Firewall.

To disable stateless FTP filtering, open an administrative command prompt, and type the following:

netsh advfirewall set global StatefulFTP disable

Remote path

The remote path on the FTP server where files need to be uploaded to (if different from the default log in directory of the FTP server). Click the Browse button to select, create or delete a folder on the remote host (if user rights allow it).

Filename

JobInfo substitution is supported by enclosing the name of the JobInfo, which holds the filename, with hash marks e.g. #Filename#. The value of the JobInfo will then be used as the filename when transferring Fixed filenames like test.txt is also supported.

Transfer mode

Active mode, passive mode and Extended Passive mode are supported.

In **active mode,** the FTP client opens a dynamic port, sends the FTP server the dynamic port number on which it is listening over the control stream and waits for a connection from the FTP server. When the FTP server initiates the data connection to the FTP client, it binds the source port to port 20 on the FTP server.

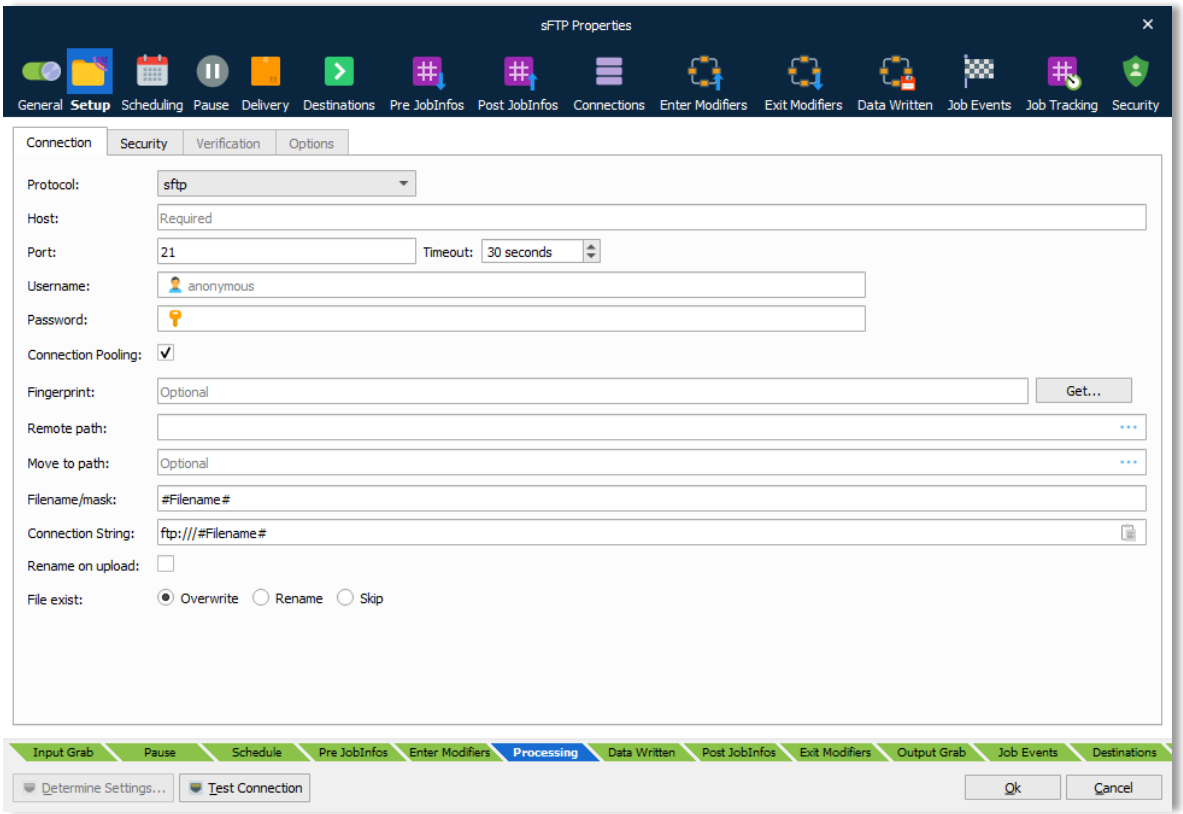
In **passive mode**, the FTP server opens a dynamic port, sends the FTP client the server's IP address to connect to and the port on which it is listening over the control stream and waits for a connection from the FTP client. In this case, the FTP client binds the source port of the connection to a dynamic port.

In **extended passive mode**, the FTP server operates exactly the same as passive mode, however it only transmits the port number (not broken into high and low bytes) and the client is to

	assume that it connects to the same IP address that was originally connected to.
Rename on Upload	Option to upload file with a temporary filename and rename when done, to avoid it being picked up before completion.
File exist	Set behaviour for uploaded file if it already exists in the selected location. **Overwrite** an existing file with contents of the uploaded file. **Rename** will keep the existing file and rename the uploaded file. Define the splitter symbol and number of digits for the counter. **Skip** existing upload if a file with same name already exists. Status for process job will be set to successful. Enable “Fail job” to set status for processed job to fail if the filename already exists.

6.4.8.2 SFTP

The SSH File Transfer Protocol (SFTP) uses a cryptographically protected connection to communicate over. Enter the port of the server into the port field if it is not port 22 (default for SFTP).

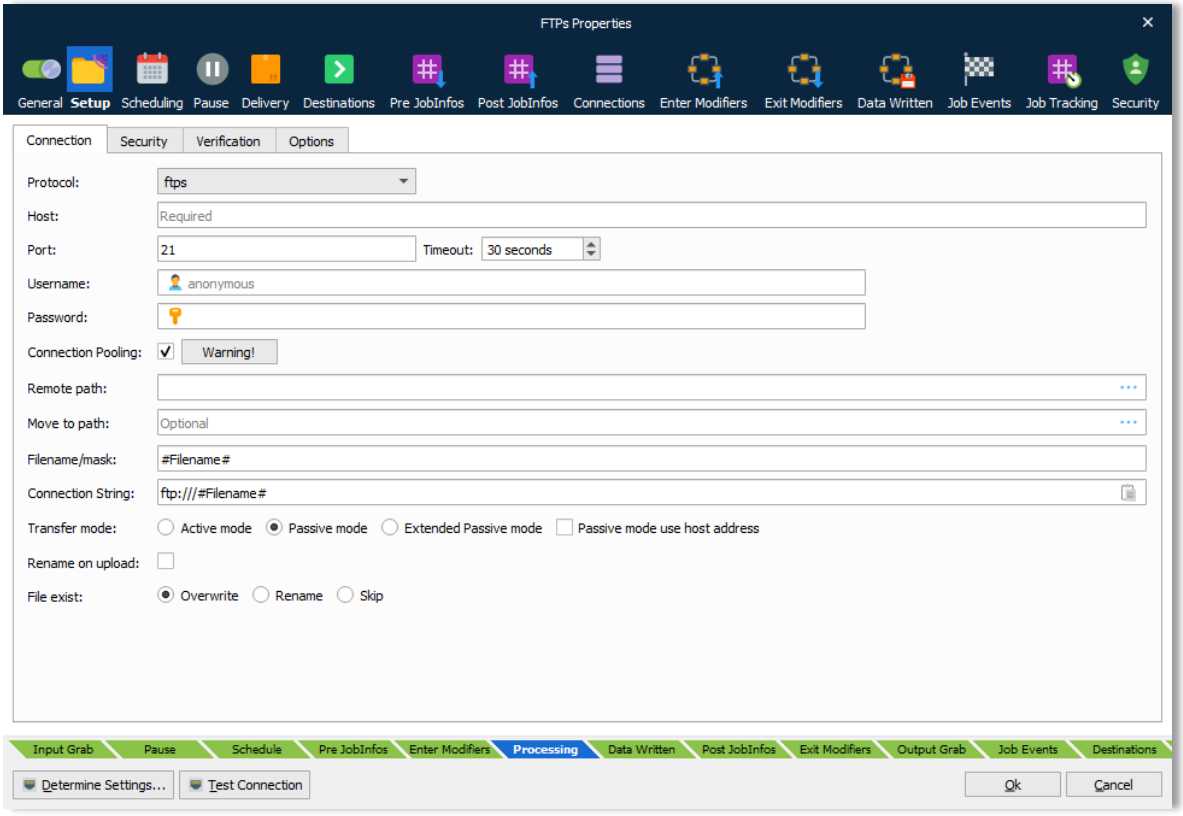


The SFTP protocol has almost identical settings to the FTP protocol, expect for the following parameters:

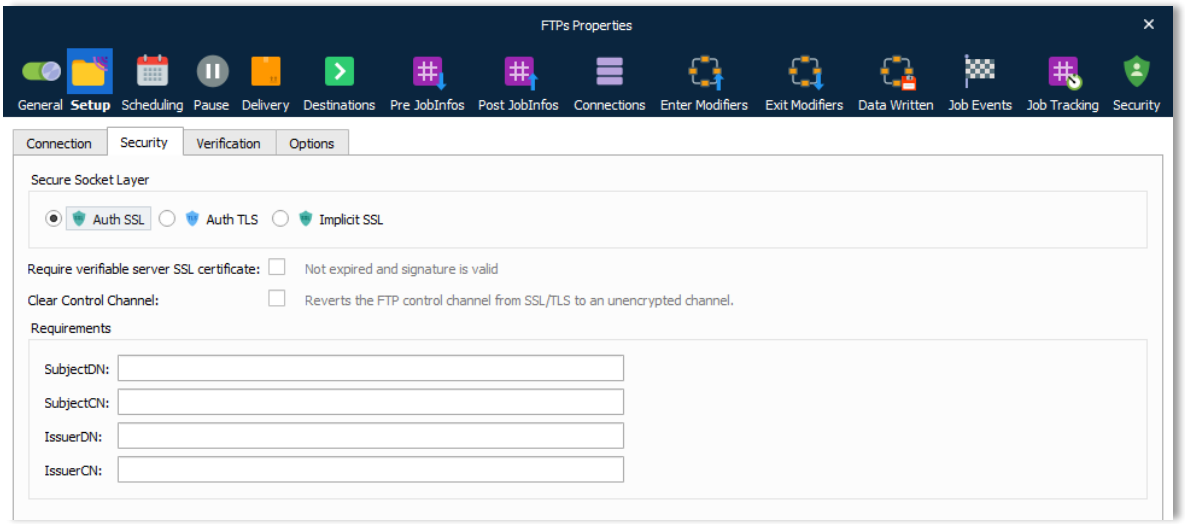
Transfer mode	Not available for SFTP.
Fingerprint	Before establishing a connection, the SFTP server sends an encrypted fingerprint of its public host keys to ensure that the SFTP connection will be exchanging data with the correct server. Once you have established a connection to an FTP server and are sure that it is the correct server, you should save the fingerprint information locally. This enables you to check the fingerprint information against the data you have saved every time you establish a new connection to ensure that no one is between you and the server.

6.4.8.3 FTPS

To enable the Security tab, FTPS must be selected as the protocol. Enter the port of the server into the port field if it is not port 990 (default for FTPS).



The Security tab contains the following settings:



Secure Socket Layer

FTPS (SSL/TLS) is available in two incompatible modes. If using explicit FTPS, the client connects to the normal FTP port and explicitly switches into secure (**SSL/TLS**) mode with **AUTH TLS**, whereas **Implicit SSL** is an older style service that assumes SSL/TLS mode right from the start of the connection (and normally listens on TCP port 990, rather than 21).

If a **verifiable server SSL certificate** is required it must be activated. Ensure that certificate is not expired and the signature is valid.

Using **Clear Control Channel** enables the Firewall to open the correct ports.

Requirements

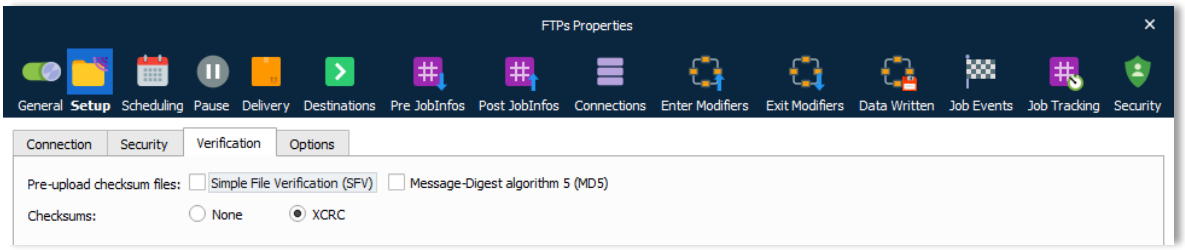
SubjectDN: Identifies the DN of the server's certificate

SubjectCN: The certificate owner's common name

IssuerDN: Identifies the CA that issued the certificate

IssuerCN: The certificate issuer's common name

6.4.8.4 Verification



Pre-upload checksum files

A checksum file can be sent first and used for real-time error checking as subsequent files are uploaded. This is a feature or

script which the FTP server must support. Lasernet does not currently handle any errors that might occur here.

Alternatively, XCRC can be enabled if the server supports it. Files are automatically checked and fixed while uploading, guaranteeing correct delivery.

6.4.8.5 JobInfos

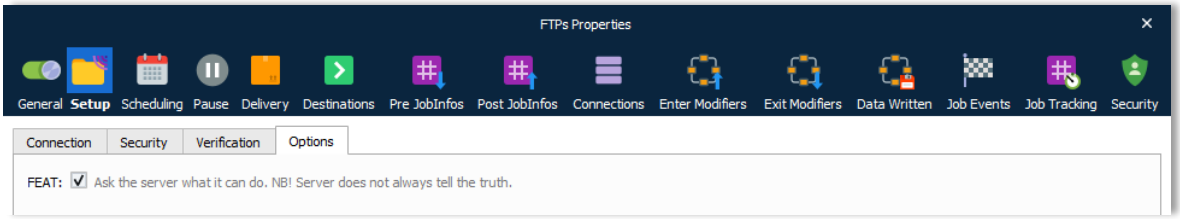
Scale

Filename associated with the SFV file. The ".sfv" file extension indicates a checksum file containing 32-bit CRC32 checksums in simple file verification format.

MD5Filename

Filename associated with the MD5 file. The ".md5" file extension indicates a checksum file containing 128-bit MD5 hashes in md5sum format.

6.4.8.6 Options



FEAT

Active FEAT if you need to ask the server for additional commands that it supports outside the basic FTP protocol defined in RFC 959.

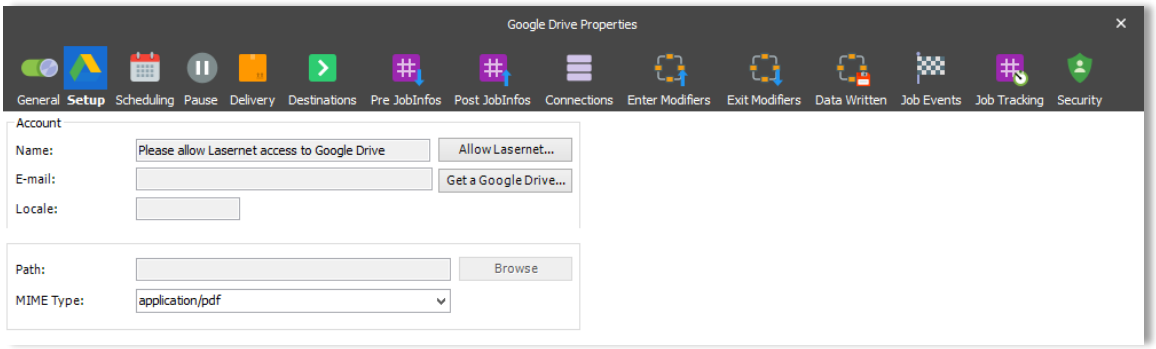
6.4.9 Google Drive

Google Drive lets you store and access your files anywhere.

Important: This module is deprecated in Lasernet and will not be supported in future versions of Lasernet.

6.4.9.1 Output

Lasernet can save files in a folder in Google Drive. Google Drive interfaces with many applications and other services like Google Docs. Since a drive can store files of different types, it is necessary to specify a MIME type when saving a file.



JobInfo substitution is supported for both the JobInfo 'Filename' and the MIME Type setting above. The path cannot be substituted as an id is actually used behind the scenes – not the path shown in the settings of the module.

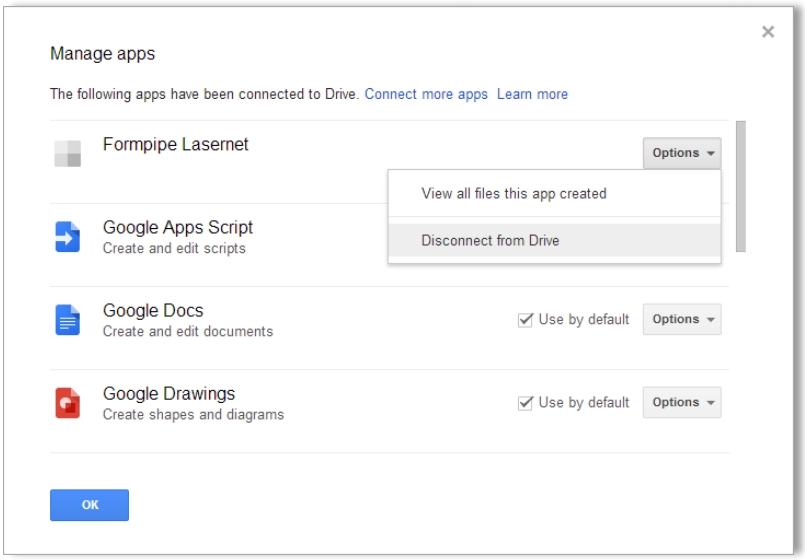
Please note that Lasernet does not currently support overwriting files in the folder. This is because a file is assigned an id upon saving. If must be overwritten, this id should be used – not the filename. Without using the id, Google Drive will create a new file with the same name but with a different id. There is no relation between these files.

6.4.9.2 Authentication

Google Drive uses OAuth 2.0 authentication. The OAuth 2.0 authorization framework enables Lasernet to obtain limited access to Google Drive by orchestrating an approval interaction between the resource owner and Google Drive. This basically means that you need to give Lasernet access to your Google Drive.

To allow Lasernet access to a Google Drive press the "Allow Lasernet" button, which launches a web browser Window where the user can login to the Google Drive account and allow Lasernet access to it. Doing this stores an encrypted token in the configuration. This token is then used to regain access to Google Drive at a later stage.

Revoking access is done via the web interface for Google Drive where it is possible to disconnect an app, in this case Lasernet, from the Drive.

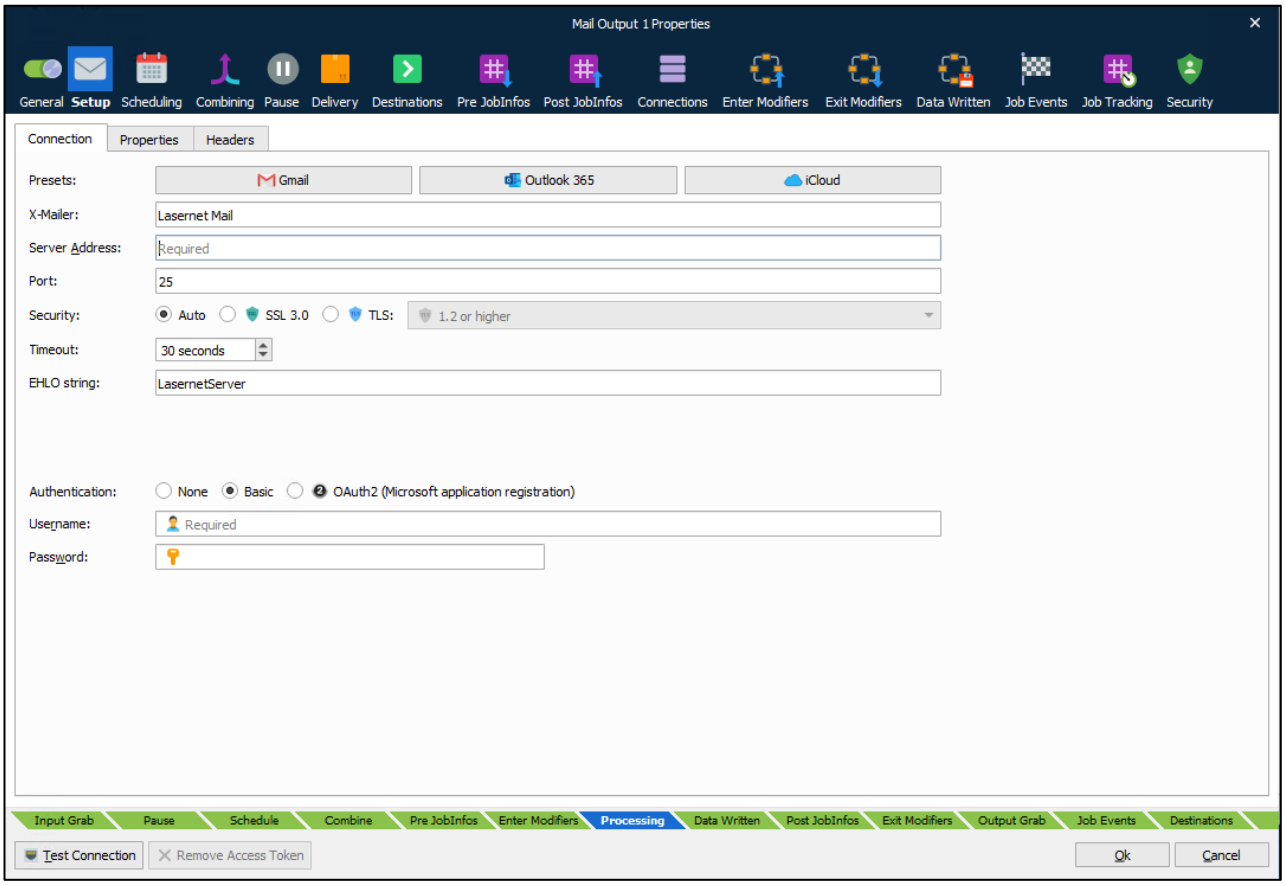


6.4.9.3 JobInfos

FileID ID of uploaded file.

6.4.10 Mail Output

Used for sending a job as email via the SMTP protocol.



6.4.10.1 General

X-Mailer — By default, the X-Mailer header identifies that Lasernet has created the email message. The field is optional and if left blank the X-Mailer will not appear in the email message.

Server Address — DNS name or IP address of SMTP server.

Port — Port number of the SMTP server.

Security — Selectable choices for security protocols are **None**, **SSL 3.0** and **TLS**. Activate SSL port and set up the port number (default 465) to provide encrypted communication and secure identification. Configure the TLS protocol to use specifically version **1.0**, **1.1**, **1.2** or **1.3** of the protocol or configure it to negotiate with the server the use of a particular version or higher (for example, **1.2 or higher**).

Timeout — If the connection cannot be established before the timeout period ends, the attempt will be abandoned. Default **Timeout** is **30 seconds**.

EHLO string — Set up the IP address for identifying the client (Lasernet) against the SMTP server.

Authentication — Select **None**, **Basic** authentication, or **OAuth2 (Microsoft application registration)** authentication.

User Name (**Basic** authentication only)	A username with access to send the e-mail via the SMTP server.
Password (**Basic** authentication only)	The password for the above username.
Tenant Domain (**OAuth2** authentication only)	Tenant domain of the app registration.
Client ID (**OAuth2** authentication only)	Client ID of the app registration.
Client secret (**OAuth2** authentication only)	Client secret of the app registration.

6.4.10.2 OAuth2 Configuration

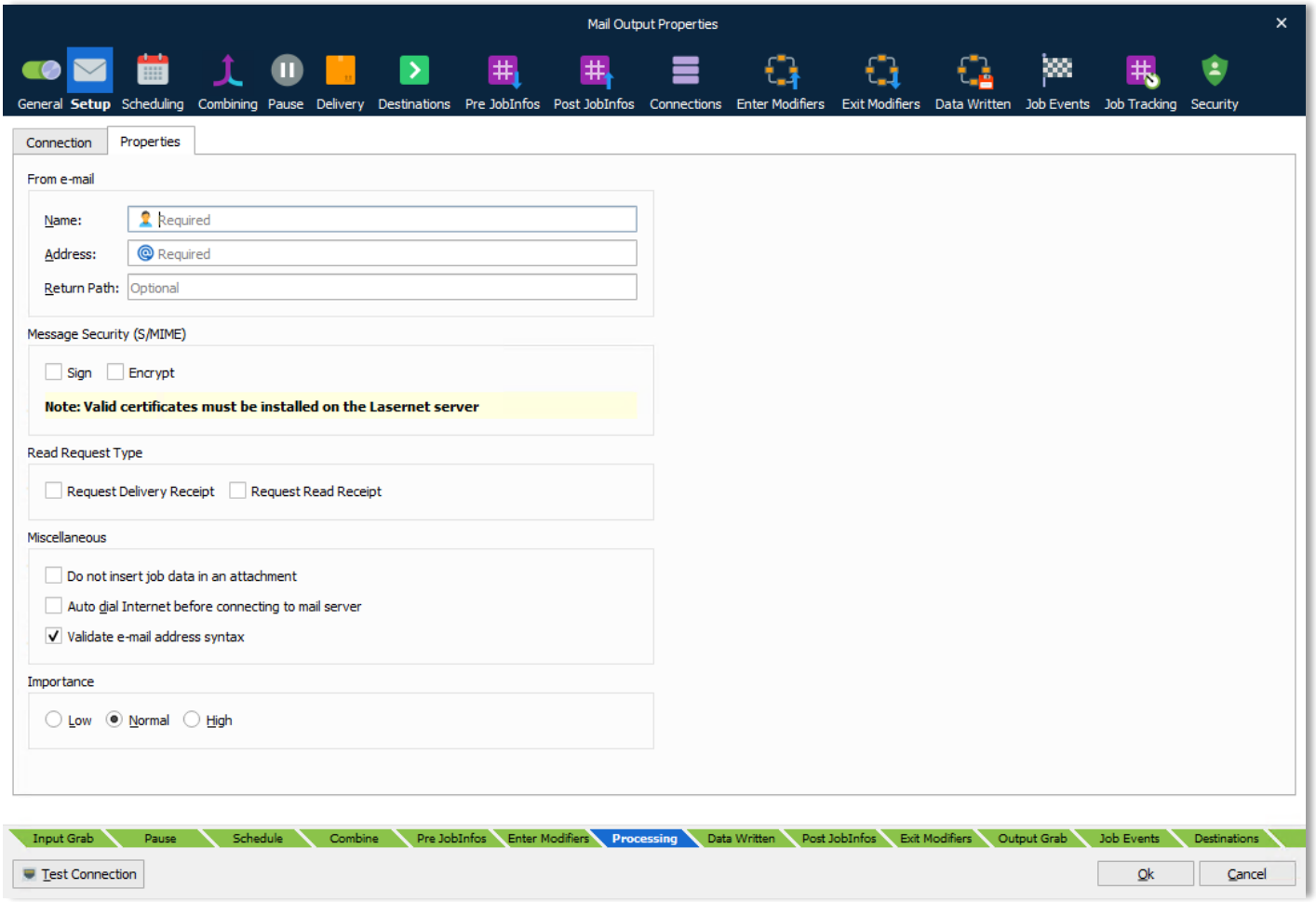
To set up OAuth2 authentication, you must complete tasks in Microsoft Entra ID and Exchange in addition to configuring this Lasernet module.

Follow this overall process:

1. In Entra ID, create an application registration for Lasernet and then grant it the following permissions:
   - For POP access: `POP.AccessAsApp`
   - For IMAP access: `IMAP.AccessAsApp`
   - For SMTP access: `SMTP.SendAsApp`
2. In Entra ID, get tenant admin consent for the Lasernet application registration to access Exchange mailboxes through POP, IMAP, or SMTP.
3. In Exchange, use the `New-ServicePrincipal` cmdlet to register the service principal.
   - **Note:** This step must be completed by an Exchange administrator.
4. In Exchange, use the `Add-MailboxPermission` cmdlet to grant the application the necessary access to specific mailboxes.
   - For example: `Add-MailboxPermission -Identity "john.smith@example.com" -User 3f2504e0-4f89-11d3-9a0c-0305e82c3301 -AccessRights FullAccess`

Note: The process above summarises the following Microsoft documentation. For full instructions and accompanying examples, go to https://learn.microsoft.com/en-us/exchange/client-developer/legacy-protocols/how-to-authenticate-an-imap-pop-smtp-application-by-using-oauth#use-client-credentials-grant-flow-to-authenticate-smtp-imap-and-pop-connections.

6.4.10.3 Properties



Name	From e-mail name
Address	From e-mail address
S/MIME: Sign	Signs the email with the S/MIME certificate associated with the **Address** in the **From e-mail** area. (This requires that certificate is imported into Microsoft Windows on the machine running the Lasernet service.)
S/MIME: Encrypt	Encrypts the email with each S/MIME certificates associated with each recipient's addresses. (This requires that the certificate for each receiver is imported into Microsoft Windows on the machine running the Lasernet service.) If not, all receiver certificates are available, no e-mail is sent.
Request Delivery Receipt	If selected, a delivery receipt will be sent to the e-mail address when the message is successfully delivered to the recipient.
Request Read Receipt	If selected, a receipt will be sent to the e-mail address when the recipient has opened the e-mail.
Do not insert job data in an attachment	If selected, job data will not be inserted as an attachment in the e-mail. Used if you only want a mail body to be presented (other attachments can still be added via the MailAttachment

	functionality – see JobInfo section for MailAttachment and MailAttachmentFileName).
Auto dial Internet	If selected, Microsoft Windows Internet connection settings will be used to dial up. It will use the default connection selected by the user on whose behalf the Lasernet service runs.
Importance	Set the category of importance to **Low**, **Normal** (default) or **High**.

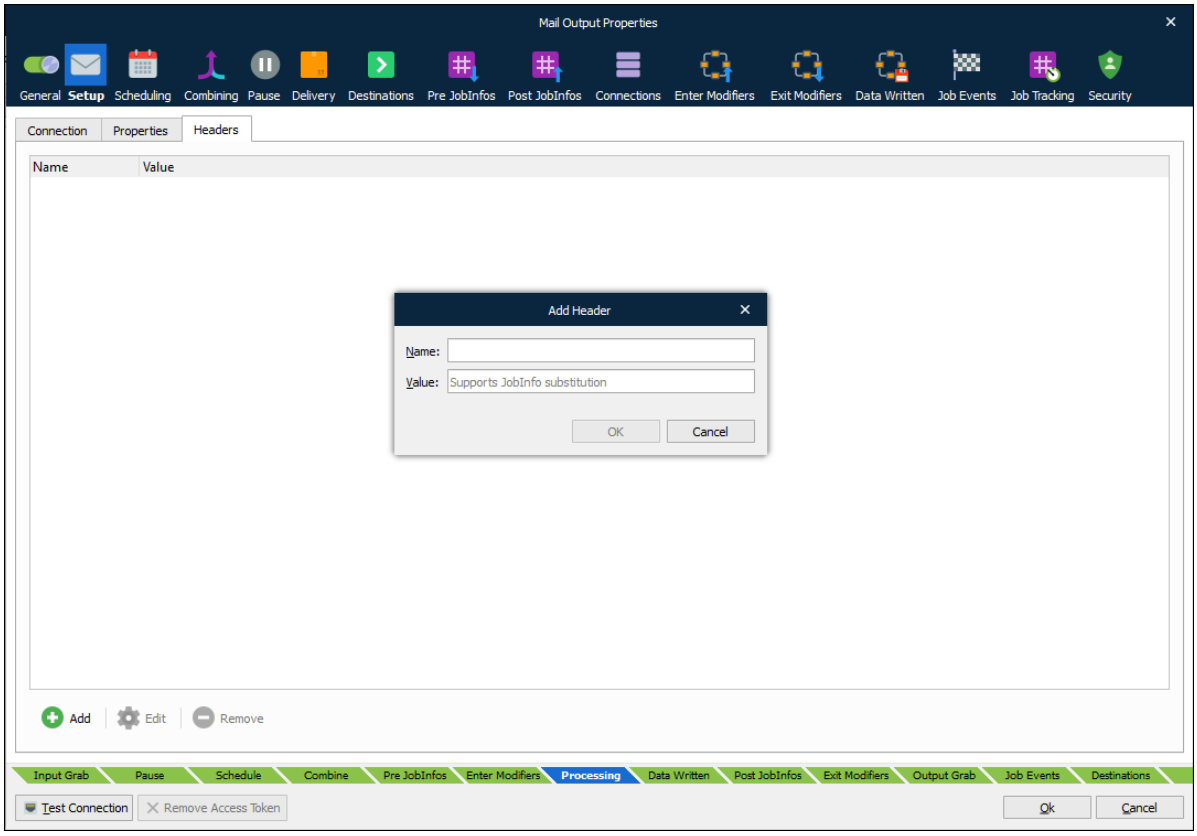
6.4.10.4 Headers

Optionally, you can custom add MIME headers to the emails sent by a Mail Output module. Some possible uses for custom headers are to enable the sending organization to troubleshoot delivery issues by examining unique codes in the header data in bounced emails, and to enable enhanced workflow visibility by enabling the organization to track emails back to their originating Lasernet process or job.

The Mail Output module will add the configured custom headers to outgoing email and give each header the specified value.

You can specify a static value for a header; for example, **Lasernet Mail**. Else you can use JobInfo substitution if you want header values to be dynamic. For example, you could enter a JobInfo reference (like **#InvoiceNumber#**) as the header's **Value**.

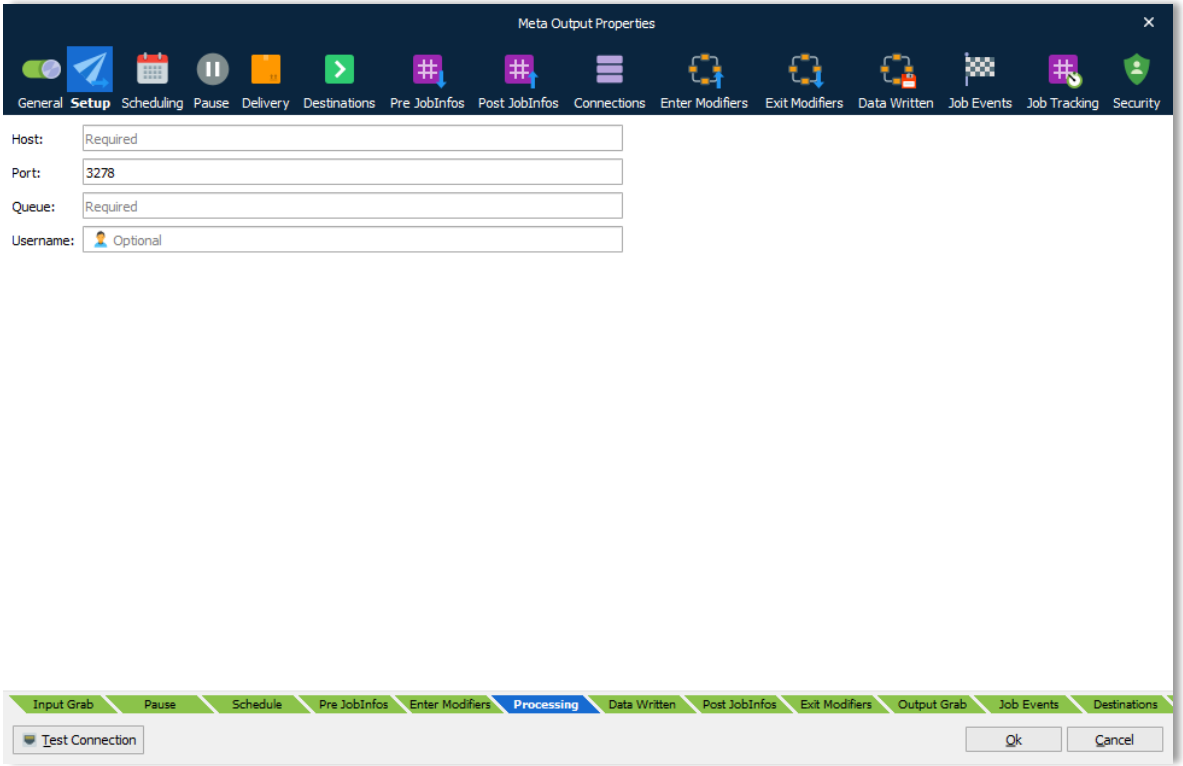
If a mail is bounced back to the sender and is read by Lasernet's Mail Input module, Lasernet creates JobInfos that have names and values corresponding to the custom headers in the bounced email.



Name	The custom header to add to email.
Value	The value for the header. Supports JobInfo substitution.

6.4.11 Meta Output

The Meta Output module send jobs to Lasernet Meta. You only have to define a host name, a port number, the name of the Meta queue and a username (optional) to send the job to.



Please refer to the “Lasernet 10 – Meta” manual for more information.

Host	Name of computer running Lasernet Meta. The field has JobInfo substitution support. Type the name of a JobInfo, enclosed by hash marks e.g., #NameOfHost#, and the value of the JobInfo will be substituted with the host name. IP addresses and localhost as computer name are not supported.
Port	The proxy port number (default 3278) to set up a connection between Lasernet Meta Output (server) and Lasernet Meta (client).
Queue	Name of Meta queue to send the job to.

Username	Name of user to retrieve document. This username is only required/supported if users are running Lasernet Meta in a Citrix or a Windows Terminal Server environment.

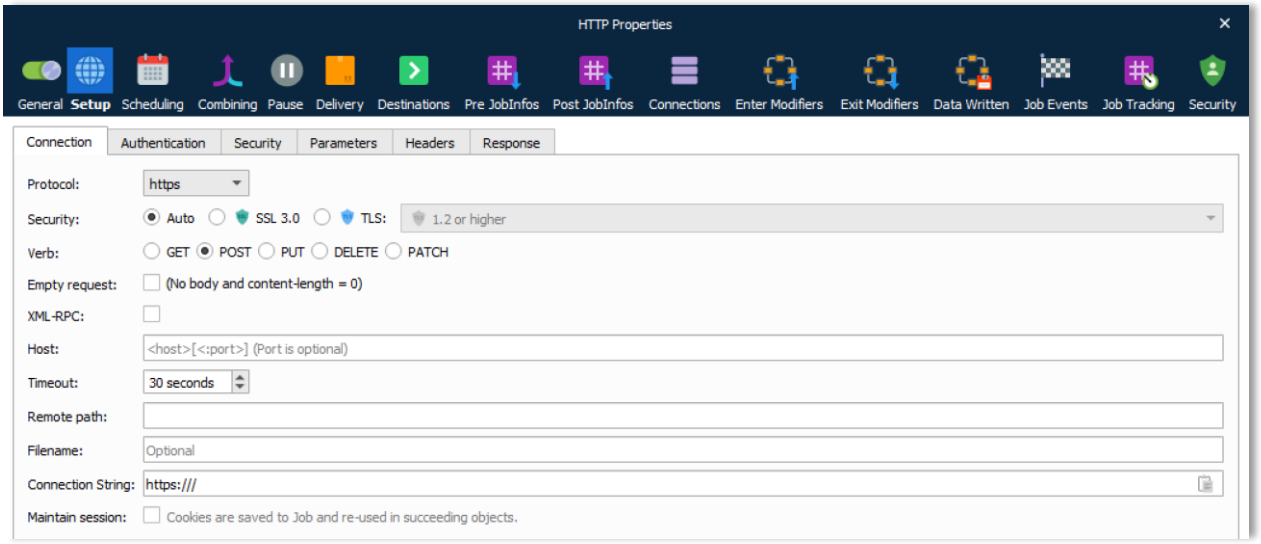
6.4.11.1 JobInfos

A list of JobInfos overrules the connection settings.

MetaHost	Name of computer running Lasernet Meta.
MetaPort	Listen port for Lasernet Meta clients.
MetaQueue	Name of Meta queue to send job to.
MetaUsername	Name of user to retrieve document.

6.4.12 HTTP

Used for sending jobs to an Internet URL destination such as a HTTP or HTTPS server.



6.4.12.1 Connection

Protocol

Supported protocols are HTTP and HTTPS.

Security

Only visible for HTTPS protocol. Selectable choices for security protocols are Auto, SSL 3.0 or TLS. Activate SSL port and set up the port number (default 465) to provide encrypted communication and secure identification and the TLS protocol to select between Auto, 1.0, 1.1, 1.2 and 1.3 or highest if negotiated with server.

Verb

The GET method is used to retrieve whatever information is identified by the Request-URI.

The POST method is used to request that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI in the Request-Line.

The PUT method requests that the enclosed entity be stored under the supplied Request-URI.

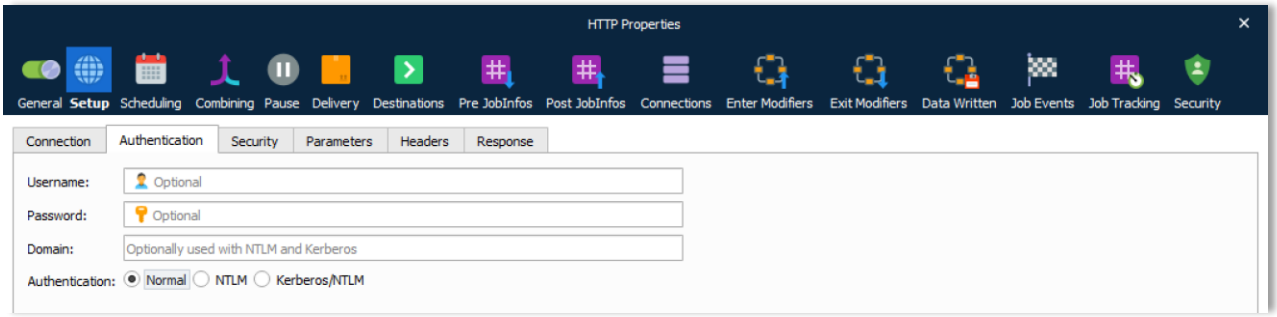
The DELETE method requests that the origin server delete the resource identified by the Request-URI (JobData will not be send when using the DELETE method, only the resource is deleted).

The PATCH method is used to update partial resources.

Empty request

To upload no content in body, activate this setting, and the job will not be inserted automatically.

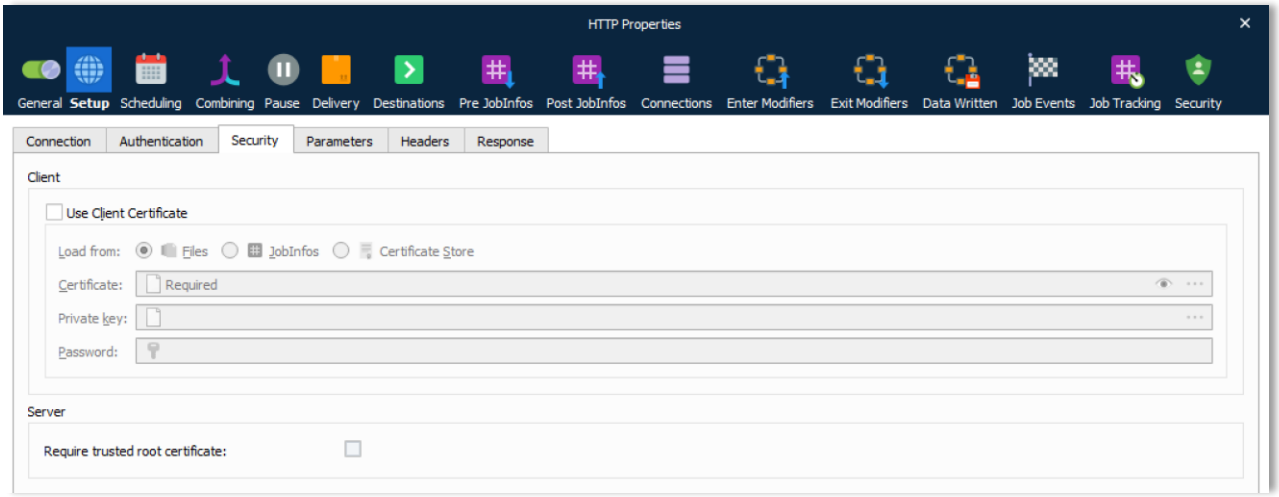
XML-RPC	Activate to use HTTP as the transport and XML as the encoding.
Host	URL for HTTP. Protocol type or remote path should not be included as part of the URL.
Timeout	If the connection cannot be established by the timeout has been reached, the attempt will be abandoned. Default timeout is 30 seconds.
Remote path	The remote path on the HTTP server where files need to be uploaded to, if different from the default login directory.
Filename	Wildcards are supported for returning one or more files to retrieve.
Maintain session	Activate to save cookies to Job and re-use in succeeding objects.



6.4.12.2 Authentication

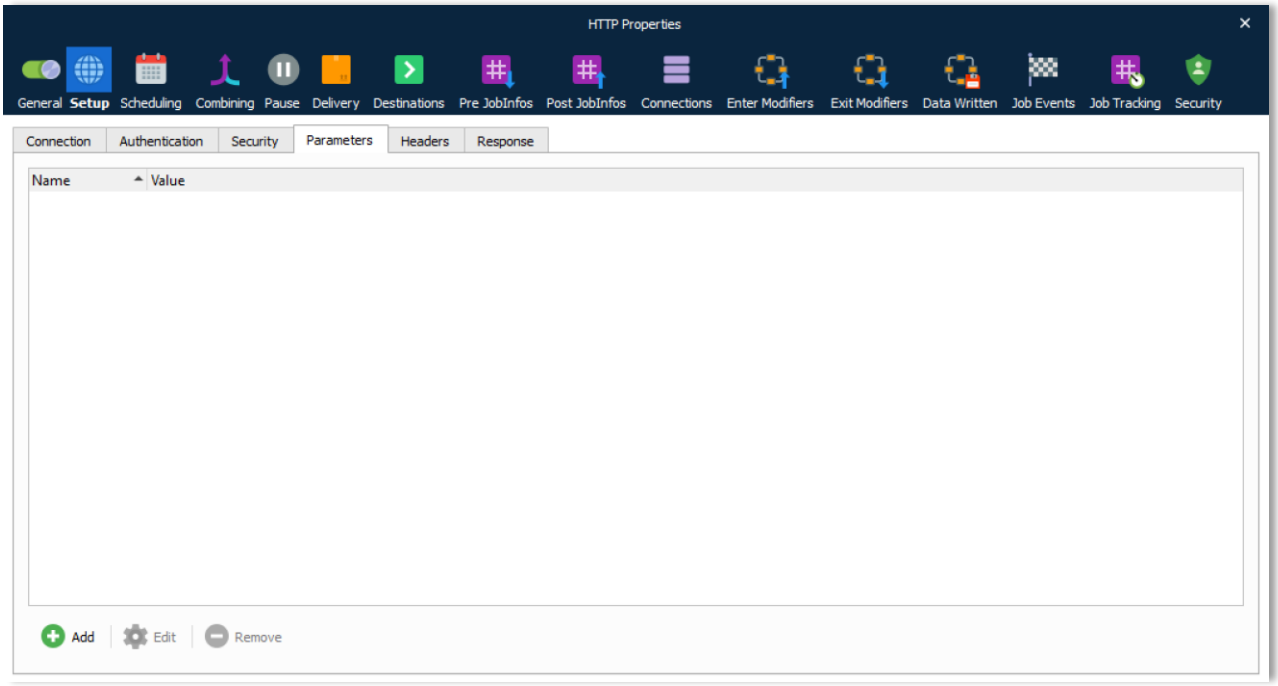
Username and password	Set username and password if basic authentication is required.
Domain	Used with NTLM and Kerberos authentication.
Authentication	Supported types are Normal authentication, NTML and Kerberos/NTLM.

6.4.12.3 Security



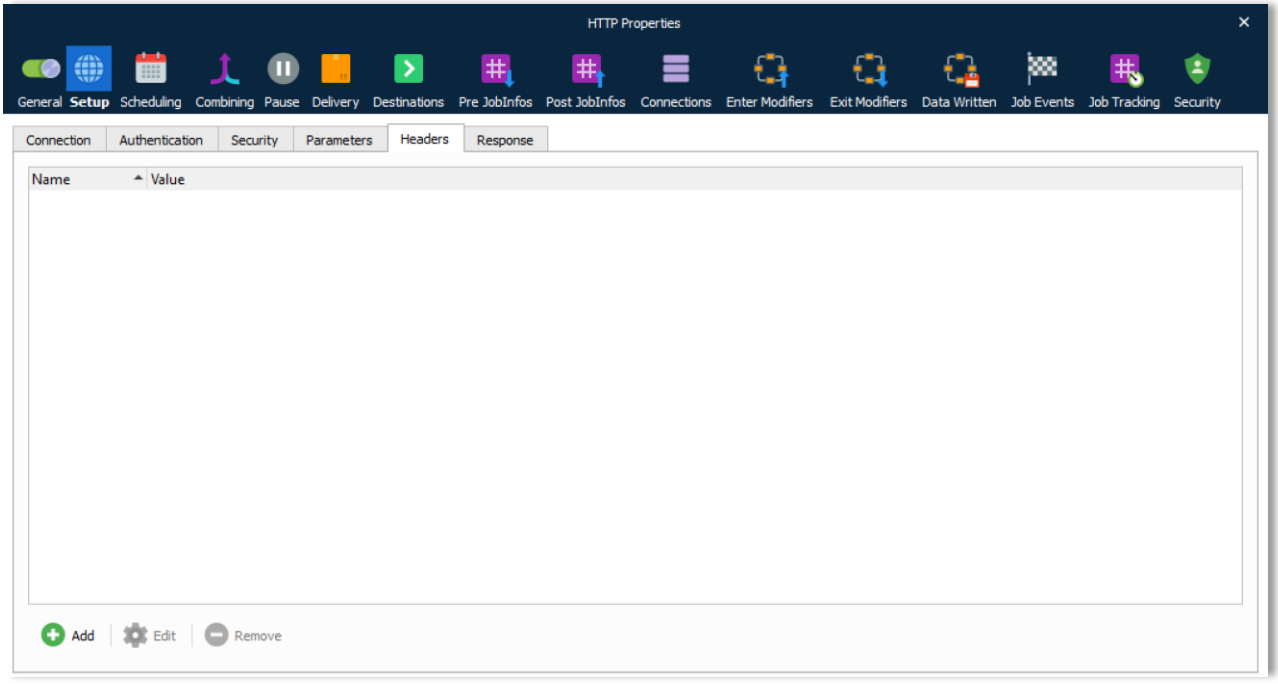
Use Client Certificate	For HTTPS it is usually not necessary to specify a client certificate and private key. If required you must activate this setting,
Load from	Select to load the client certificate from: 1. File with one of the extensions for X.509 certificates 2. Retrieve the certificate and private key from a JobInfo 3. Load the certificate from a store
Client Private Key	To be authenticated select a private key for the client.
Private Key Password	Set the password for the private key.
Subject Common Name (CN)	When enabled you must set a CN to match the subject for the host being authenticated.
Require trusted root certificate	Enabling this option will perform a range of verifications of the certificate used for HTTPS: 1. Validation of the certificate. 2. Expiration of any certificate in the chain. 3. Root certificate is trusted.

6.4.12.4 Parameters



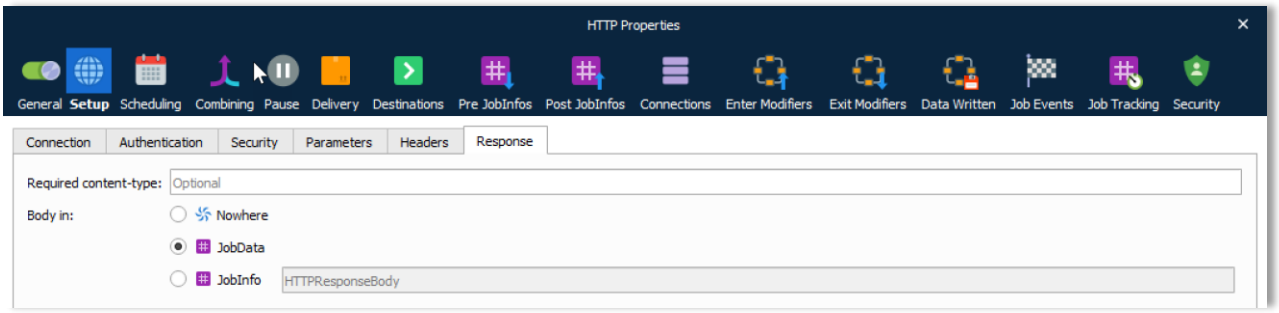
Parameters Add a set of parameters with values to the HTTP request.

6.4.12.5 Headers



Headers	Add a set of headers with values to the HTTP request.

6.4.12.6 Response



Required content-type	Content-type required for document.
Body in	After receiving and interpreting a request message, the server responds with an HTTP response message. You can define where to insert the HTTP body. Select **Nowhere** to store it nowhere, **JobData** if you are going to use it as job data in another module, or **JobInfo** if you want to insert the value into a specific JobInfo.

6.4.12.7 JobInfos

The HTTP Output module sets these JobInfos.

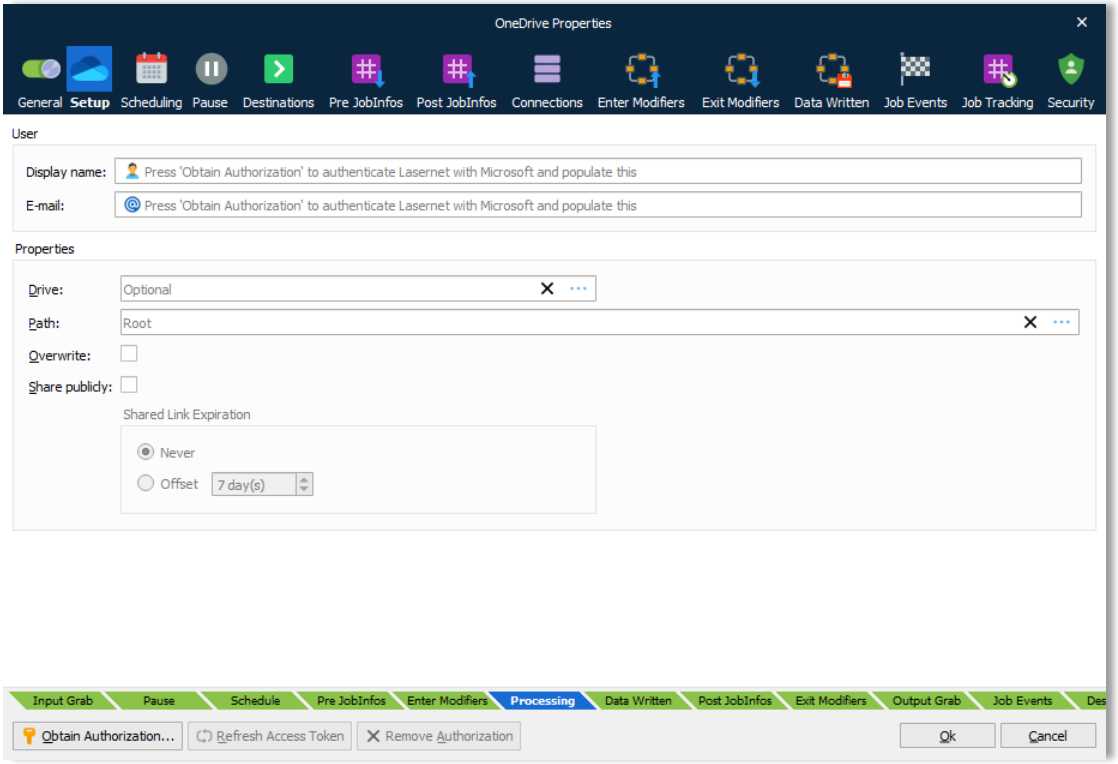
HTTPHeaderFieldName	JobInfo list containing field names for the HTTP response headers.
HTTPHeaderFieldValue	JobInfo list containing field values for the HTTP response headers.
HTTPHeader*	The headers described by **HTTPHeaderFieldName** and **HTTPHeaderFieldValue** are also stored in individual dedicated JobInfos. For example, **HTTPHeaderContent-Length**.
HTTPStatusCode	Status code for HTTP response.
HTTPStatusText	Status text for HTTP response.

6.4.13 OneDrive

Easily store and share documents in the cloud. Microsoft OneDrive works with Office, so it's easy to create, edit, and share your documents.

6.4.13.1 Output

Save files in a folder in Microsoft OneDrive.



6.4.13.2 User

1. Click **Obtain Authorization** in the OneDrive dialog box to create a new connection.
2. You get redirected to the Microsoft website to **authorize OneDrive for Lasernet**.
3. **Enter** your Microsoft credentials.
4. **Authorize** the access request.
5. The authentication window closes automatically and your **Display name** and **E-mail** are added to the dialog.
6. In the Properties you can set your **Drive** and **Path** for where to poll for incoming files.

Doing this stores an encrypted token in the configuration. This token is then used to regain access to the OneDrive later.

6.4.13.3 Properties

Select the **Drive** and **Path** for where to upload the files in your OneDrive file structure.

Activate **Overwrite** to allow that a file, that already exist, is overwritten during upload.

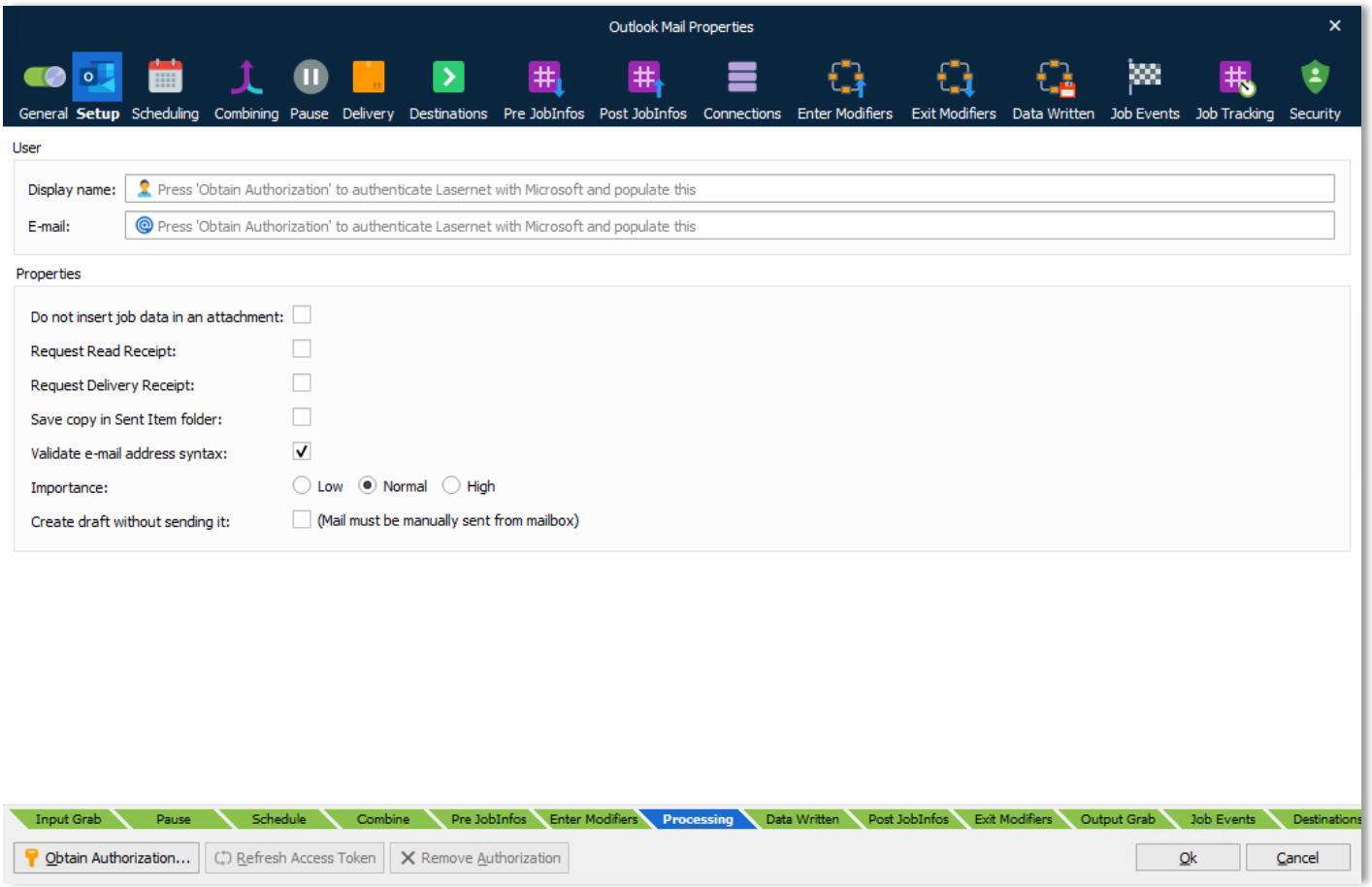
6.4.13.4 JobInfos

OneDriveSharePublicly	Set this JobInfo as a **Boolean** value to overwrite the setting for **Share publicly** property.
OneDriveURL	URL to publicly shared file returned after upload. The JobInfo is only set when **Sharing publicly** is enabled in the GUI.
OneDriveURLExpiration	This JobInfo overwrites the offset of the **Shared Link Expiration** defined in the OneDrive module. A expiration time for a shared document must be defined in the format **yyyy-MM-ddTHH:mm:ssZ**.

6.4.14 Outlook Mail

The Outlook Mail module integrates directly with Office 365 to send emails. It is an alternative to the Exchange module using Exchange Web Services (EWS) introduced by Microsoft in Exchange Server 2007. EWS will longer receive feature updates, while the service will continue to receive security updates and certain non-security updates.

Microsoft has announced that EWS will still be available and supported for use in production environments. However, we recommend the Outlook Mail module, using Microsoft Graph API, to Exchange Online data.



6.4.14.1 Obtain authentication

Click **Obtain Authorization** to access the account for your organization and to autofill the **Display name** and **E-mail**.

The Lasernet Outlook Mail module will obtain authorization to:

- Maintain access to data you have given it access to
- Send mail on behalf on others or yourself
- Sign you in and read your profile

6.4.14.2 General

Display name	Display name for the account (click **Obtain Authorization** to autofill this field).
E-mail	E-mail address for the account (click **Obtain Authorization** to autofill this field).
Do not insert job data in an attachment	If selected, an e-mail will be sent without including JobData as an attachment. EWS server will as report a non-critical error log message when sending emails without attachments: Unable to read MimeContent; This operation can't be performed because this service object doesn't have an Id.
Request Read Receipt	If selected, a receipt will be sent to the e-mail address when the recipient has opened the e-mail. **Note:** The JobInfo MailRequestReadReceipt must be set to true (1) to overrule default behaviour.
Request Delivery Receipt	If selected, a delivery receipt will be sent to the e-mail address when the message is successfully delivered to the recipient. A JobInfo overrides the default behaviour. **Note:** The JobInfo MailRequestDeliveryReceipt must be set to true (1) to overrule default behaviour.
Save copy in Sent Item folder	If selected, a copy of the e-mail will be stored in the Sent Item folder in your email client.
Validate e-mail address syntax	If selected, the email syntax in MailTo, MailCC and MailBCC will be validated before sending the email to the mail server. If the syntax is not valid, the job will fail. To prevent a job from failing the setting must be de-activated. This will allow mail addresses to contain syntax used for external fax systems connected to mail servers. Comma and semicolon are valid as separators if MailTo, MailCC or MailBCC contain multiple addresses.
Importance	Set the category of importance to Low, Normal (default) or High.
Create draft without sending it	If selected, a draft will be created to manually open and edit the e-mail before sending from mailbox.

6.4.14.3 Headers

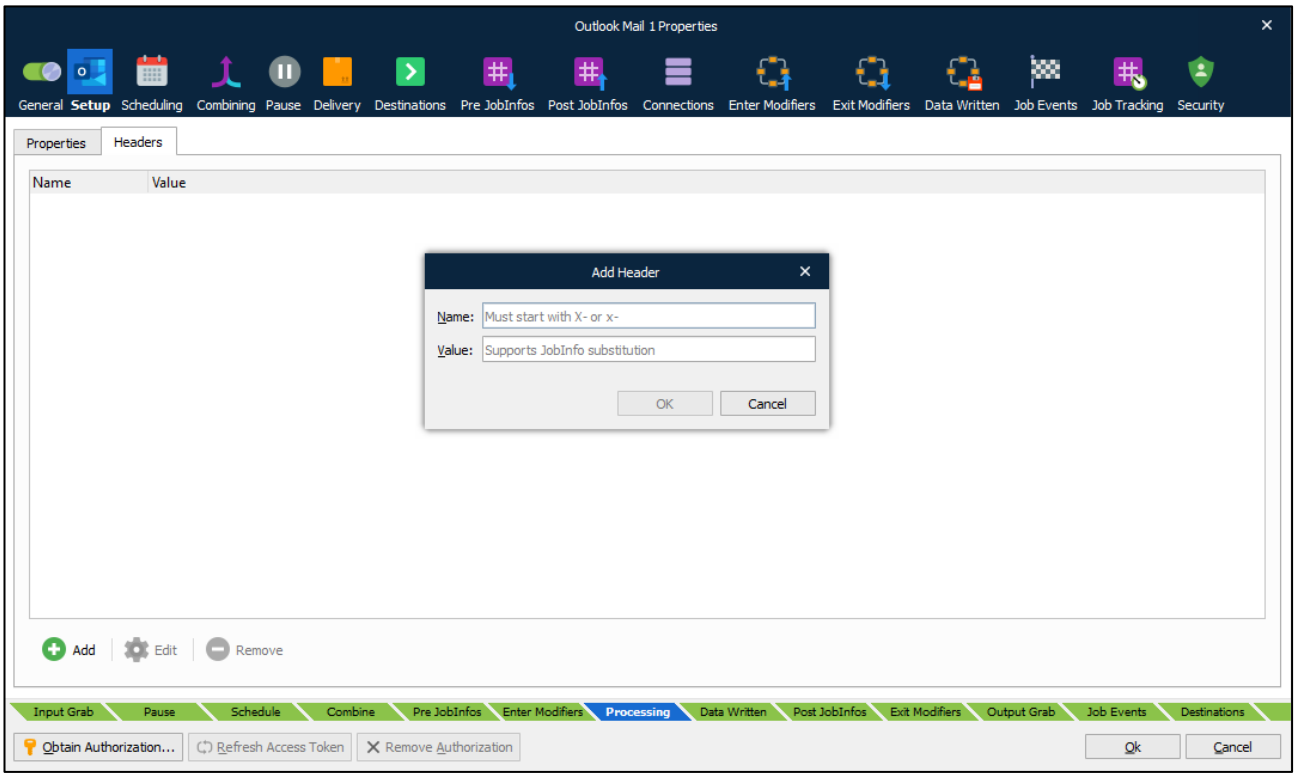
Optionally, you can add custom MIME headers to the emails sent by an Outlook Mail Output module. Some possible uses for custom headers are to enable the sending organization to troubleshoot delivery issues by examining unique codes in the header data in bounced emails, and to enable enhanced workflow visibility by enabling the organization to track emails back to their originating Lasernet process or job.

You can add up to five custom headers.

The Outlook Mail Output module will add the configured custom headers to outgoing email and give each header the specified value. A custom header's **Name** must start with **X-** or **x-**; for example, **x-Invoice-Number**.

You can specify a static value for a header; for example, **Lasernet Mail**. Else you can use JobInfo substitution if you want header values to be dynamic. For example, you could enter a JobInfo reference (like **#InvoiceNumber#**) as the header's **Value**.

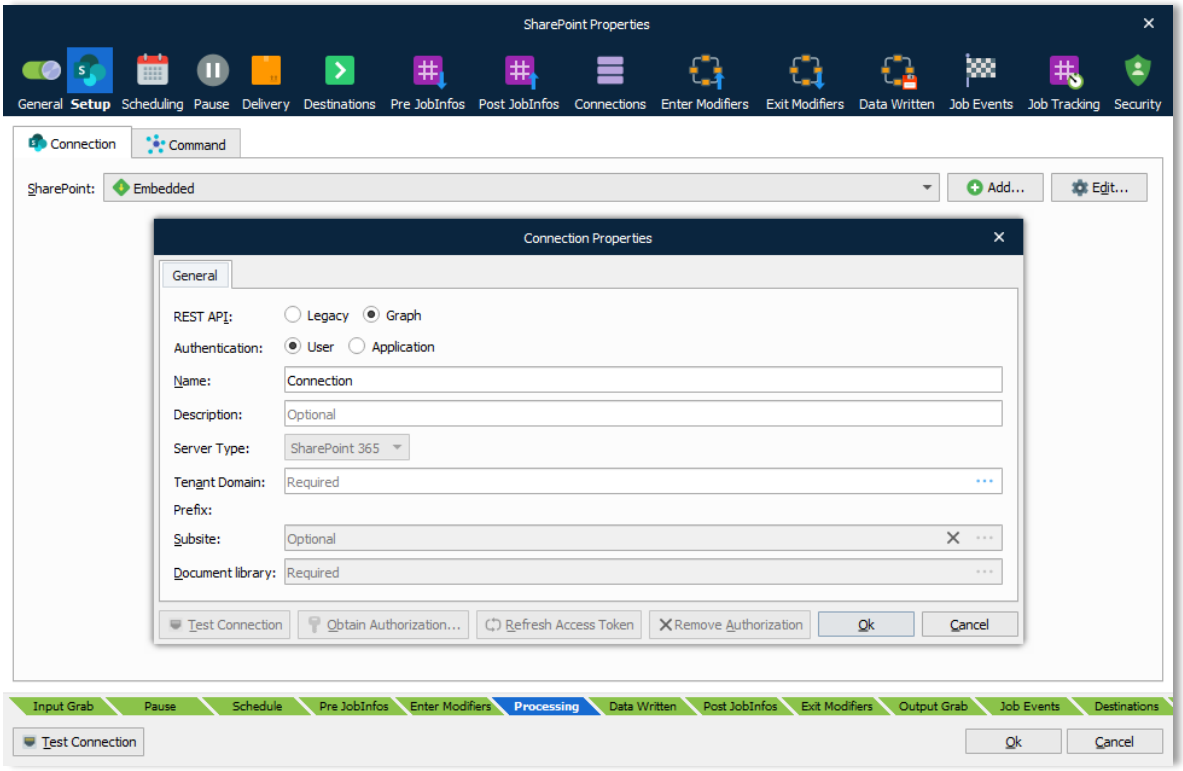
If a mail is bounced back to the sender and is read by Lasernet's Outlook Mail Input module, Lasernet creates JobInfos that have names and values corresponding to the custom headers in the bounced email.



Name	The name of the custom header added to email must start with **X-** or **x-**.
Value	Set a value for the header. Supports JobInfo substitution.

6.4.15 SharePoint

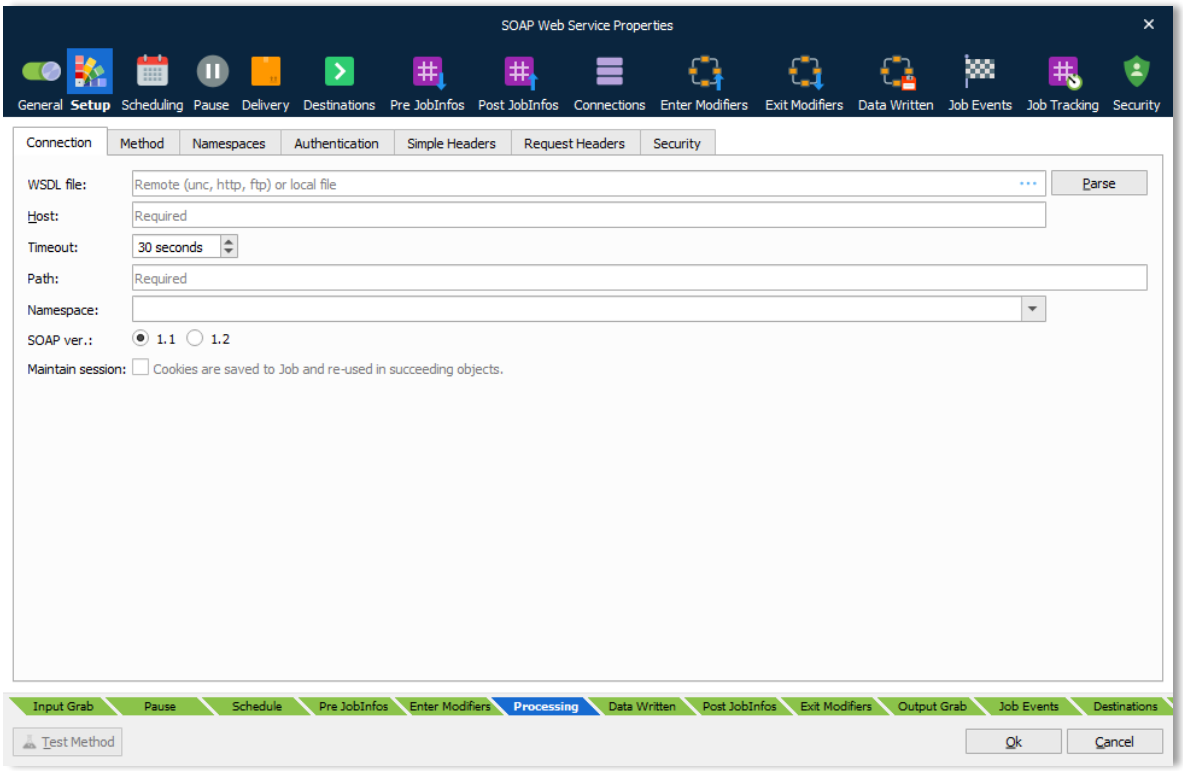
This module is used for connecting to a SharePoint server.



Please refer to the “Lasernet 10 – SharePoint” manual for more information.

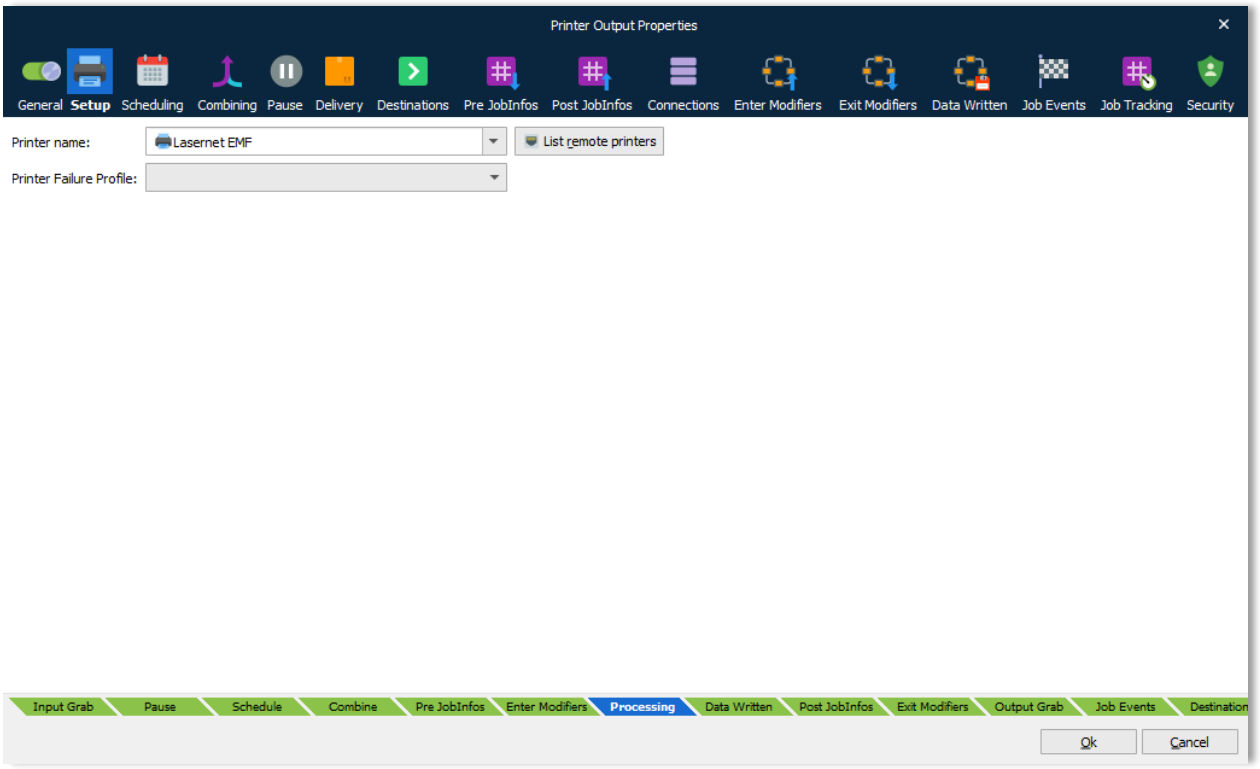
6.4.16 SOAP Web Service

The SOAP Web Service Output module can be used in the same way as the SOAP Web Service Input module to retrieve or send data to a Web Service.



6.4.17 Printer Output

Used for printing documents to a local or network printer. This module work for on-premises only and for best performance connected to a local printer, since latency must be expected to a printer shared on a network.



Note: You can read more about the alternative **Printer Service module** and stand-alone application, in the guide for the **Lasernet Printer Service**, which is a solution to manage your custom printer settings remotely and deploy these to local printers, or anywhere globally using Microsoft Azure cloud services.

In the **Printer Output module**, you specify what Windows printer name to print to. It can be the name of a local printer or a printer shared on the network. Once a valid name is entered, the printer profiles can be setup for the output module. Printer profiles are supported for documents created in the EMF format. To learn more about printer profiles see section for *Manipulating Printer Profiles*.

The Print Output module also has a built-in functionality to auto-detect and print PDF and DOCX documents, called printer attachments, without the needs for 3rd party applications, including selection of paper source in the printer by detecting the PDF page sizes in the PDF documents. Other printer settings are selected by the default printer settings defined in the Windows printer settings.

Note: Printer Profiles are not objects and, therefore, cannot be seen in the object list when using "File - Export Objects". However, Printer Profiles are linked to the Printer Output objects for which they were created. When you export a Printer Output object, any associated Printer Profiles will be included with that object, making them available when you import the printer into another configuration.

You cannot select individual Printer Profiles under the printer object when exporting. All associated Printer Profiles will be included in the **.lnobjectx** export file.

Printer failure profile
See Section 11.4 Printer Failure Profiles

Maintaining many output printers
For easier maintenance of similar output printers, without creating the equivalent number of Printer Output modules, you can set a JobInfo *PrintToUNC* which overrules the printer name set in *Printer Setup – Printer name*. This makes it possible to use Printer Profiles for these printers.

For example, by setting the *Printer name* to \\Server1\SharedDemoPrinter and then setting the JobInfo *PrintToUNC* to \\Server2\SharedPrinter, the Printer Output module will print the job to Server2 but use the printer profiles defined for Server1.

Note: Remember that the Printer name entered in Printer Setup still counts for 1 in the number of licenses used.

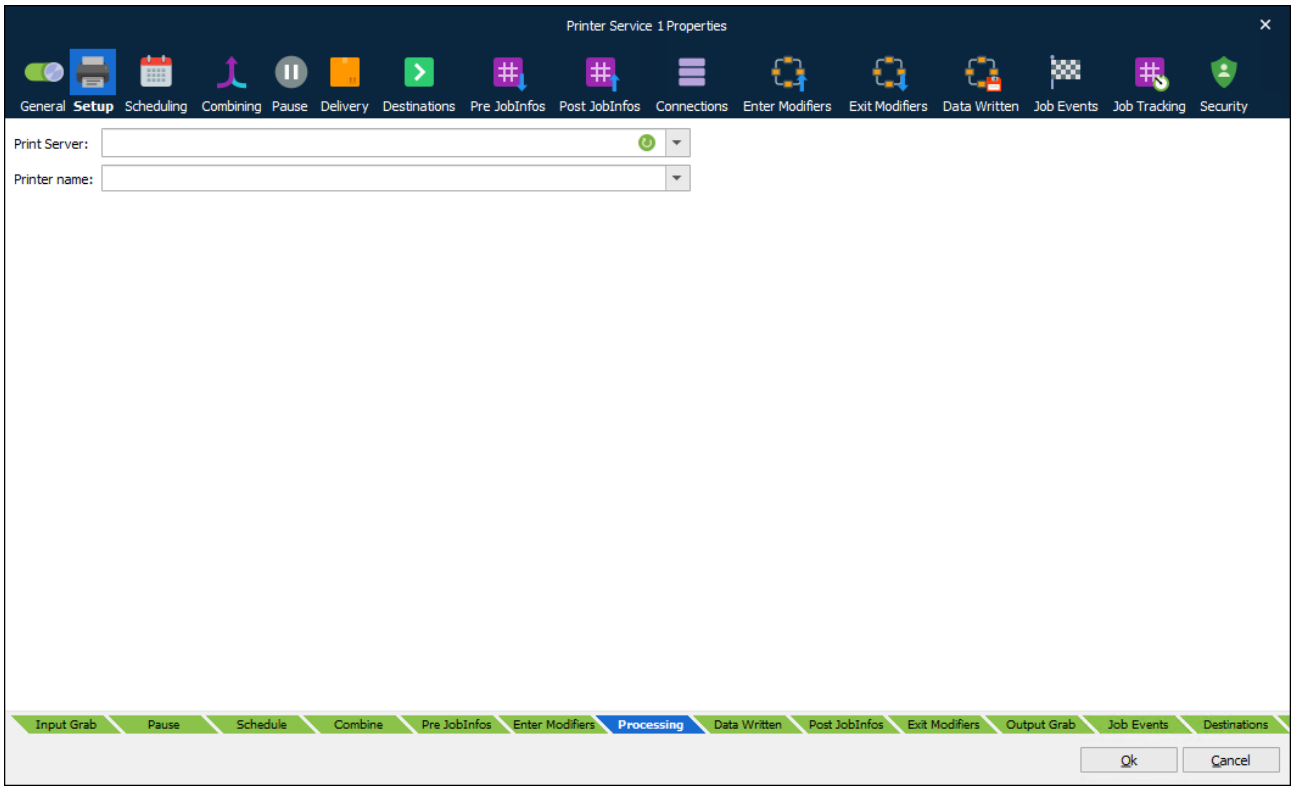
6.4.17.1 JobInfos
A list of JobInfos can overrule the printer settings:

ColorMode	ColorMode is used set monochrome or color printing. Known values are: Default, monochrome and color.
Copies	Set Copies to control the number of printed copies.
DocName	DocName JobInfo is used to control the name of the print job for the Windows Print Spooler. If not set the default value is set to “Unnamed – Lasernet document”.
DuplexMode	Duplex printing allows the printing of a sheet of paper on both sides automatically. Examples of values are: Default, Simplex, Vertical and Horizontal.
Orientation	Manage the Orientation of the paper. Example of values are Portrait or Landscape.
PaperHeight	Defines the height of the paper form.
PaperSource	Selects the Paper Source in the printer. Example of values are: Auto, Tray 1, Tray 2, Upper Tray, Lower Tray.
PaperWidth	Defines the width of the paper form.
PrintAttachment	May contain a list of documents in binary representation for printing externally. It is used together with PrintAttachmentFilename and PrintAttachmentMimeType JobInfos.
PrintAttachmentCopies	Used to set control the number of printed copies for attached documents in the format PDF and DOCX.
PrinterDriver	The PrinterDriver JobInfo is a configuration specific JobInfo often used by the EMF2RAW and Printer Output modules, if the JobInfo substitution string #PrinterDriver# is defined as the Printer Name parameter.

PrinterName	The PrinterName JobInfo is a configuration specific JobInfo, often defined in modules, if the JobInfo substitution string #PrinterName# is used as the name of the destination to the Printer Output module.
PrintFilename	If it is not empty, it is assumed that the JobData is a PDF or Word document type which should printed separately. It is used together with PrintMimeType.
PrintMimeType	Contains the mime type of the file to print. It is used together with PrintFilename JobInfo.
PrintToUNC	Directs jobs to a printer using a UNC path instead of a predetermined printer in Windows. By setting this JobInfo to \\Server\PrinterShare the Printer Output module will send the job to that share using the driver chosen in the setup.
WinPrintNotifyName	Specifies the notification contact of the print job. Windows does not allow the notify name to be set across network printers.
WinPrintUserName	Specifies the username of the print job as shown in the job list for a local printer. This can be set to just about anything. Windows does not allow the user name to be set across network printers.

6.4.18 Printer Service Output

The Printer Service module is used for connecting and sending print documents, via cloud or local network, to the Printer Service app running on a Windows Printer Server.



For more information, see the Lasernet Printer Service 10 Guide.

7 Scripts.

7.1 Expanding functionality with Scripting

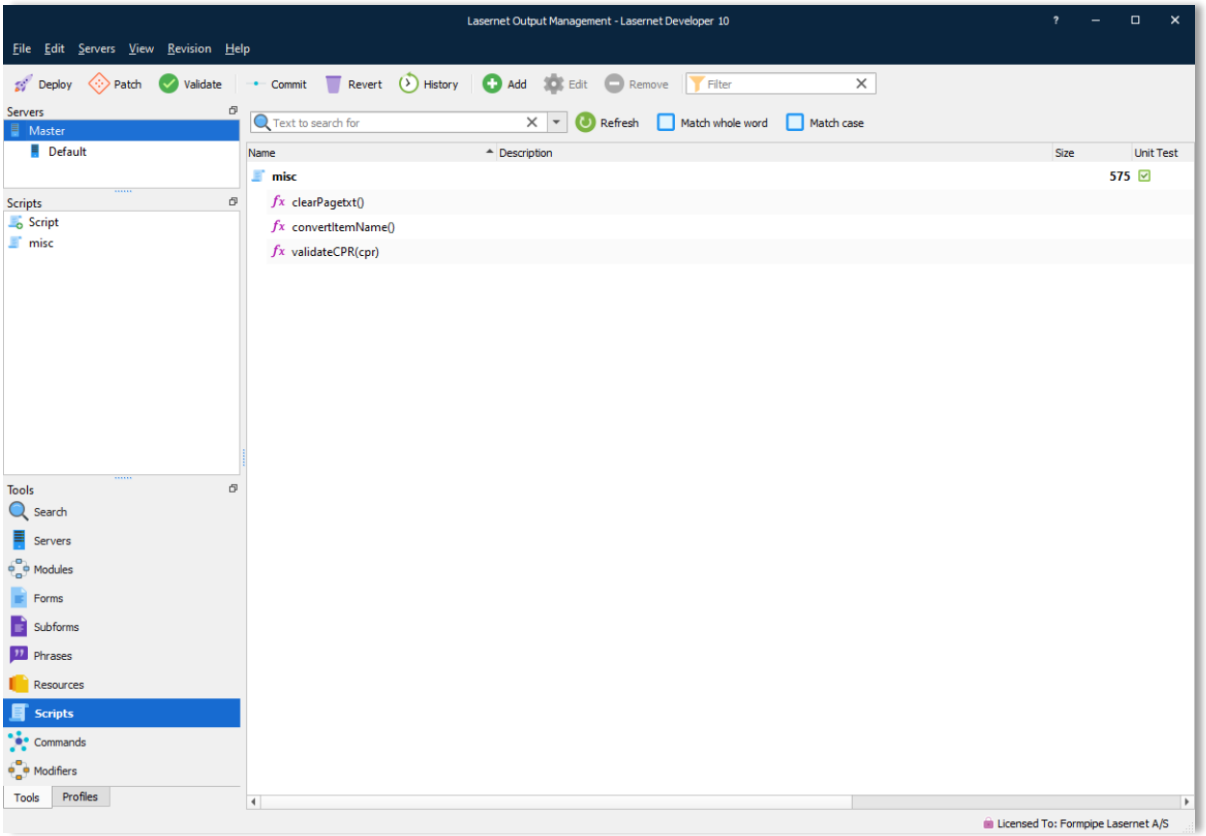
7.1.1 Introduction to Scripting

Lasernet contains a scripting framework which can be used to add advanced functionality.

Scripts are typically used for checksum calculations, conditional overlays and conditional printer profiles, though it is possible to use scripting to develop and extend any aspect of your Lasernet solution.

This introduction to scripting in Lasernet will focus on practical examples that are used in many setups. Some of the terms and explanations in this section may appear complex if you are new to scripting, but we recommend reading the section in full in order to familiarise yourself with the possibilities that scripting offers.

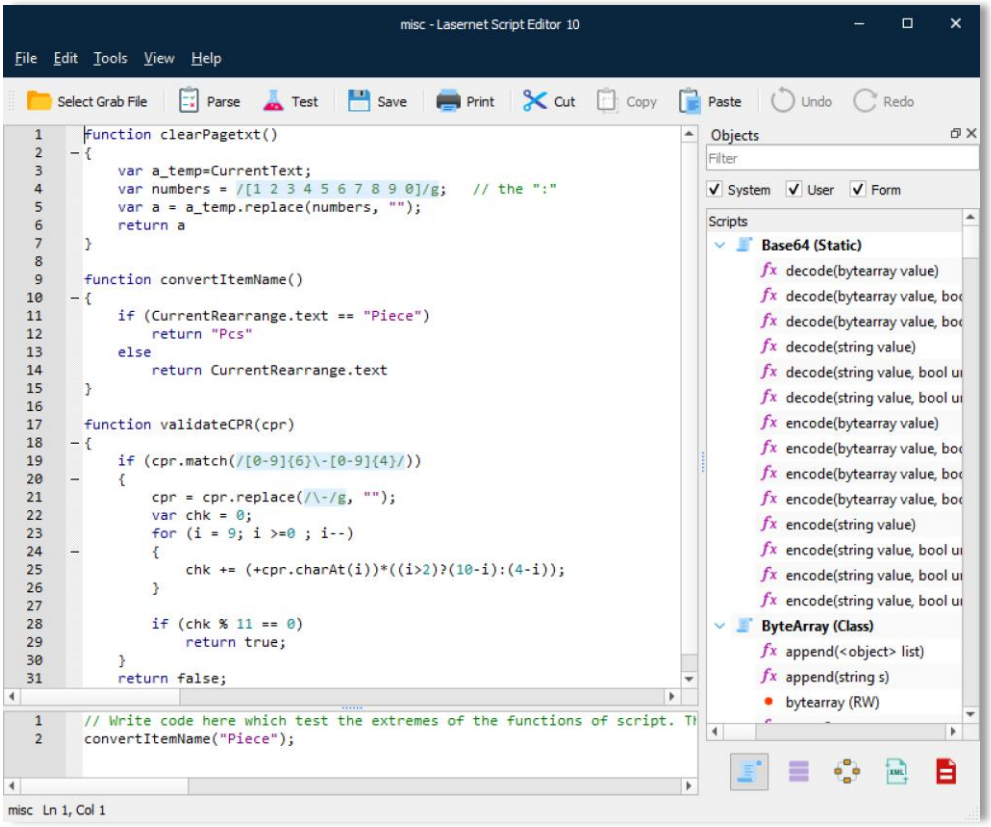
The Lasernet Developer contains a script editor with functionality for maintaining your own scripts. You access the script editor by creating a new script or opening an existing one.



To open an existing script either right-click it and choose **Edit,** or double-click it. To create a new script press the **Add** button in the toolbar or right-click and choose **Add**.

7.1.2 The Script Editor

Lasernet includes a basic script editor.



The editor supports the development of script by providing you with features for efficiently writing and editing code:

- Language awareness, automatic recognition of keywords;
- Code hints for fast addition of Lasernet functions;
- Parsing, for easy testing and debugging of your script;
- Indentation guides, for easier overview of the script structure;
- Undo/Redo.

7.1.3 Manipulating Overlays

This information applies to scripting in the Form Engine.

When producing a print, the same overlay is not generally needed for all pages. To assist script writing for manipulating overlays, Lasernet provides the ability to access the Page object.

7.1.3.1 The pages object

When the Form Engine creates text output, it creates a number of pages. Each of these pages can be accessed from within a script and the page properties can be manipulated. It is also possible to clear and/or add overlays on each page. The pages are available in an array called pages, where the first page is named pages[1] and so on. There is one item for each page in the array. The example below shows one way of adding an overlay to the first page:

```
pages[1].addOverlay(”SomeOverlay.lnemf”, true);
```

The page object is referred to by the variable ‘pages’, which makes it possible to access the page array. The value, [1], signifies the first page is required, against which the command addOverlay is appended. This method is used for instructing Lasernet to add an overlay to that particular page. Inside the parentheses two parameters can be specified. The first parameter is required and contains a string identifying the filename of the overlay that you wish to add to the page. The second parameter is optional and contains a Boolean (true/false) value where “true” means that Lasernet will remove any existing overlays on the page before adding the new one, and false means that Lasernet will keep any existing overlays. The default value of the second parameter is “false” and as such the command can be written like this:

```
pages[1].addOverlay(”SomeOtherOverlay.lnemf”);
```

This will instruct Lasernet to keep existing overlays on the page.

7.1.3.2 Short-cut functions

Lasernet also provides two shortcut functions for adding overlays to all the pages or to just the current page:

```
addOverlay(”NameOfTheOverlayFile.lnemf”, true);
addPageOverlay(”NameOfTheOverlayFile.lnemf”, true);
```

It is best practice to call the addOverlay function from an **On Sheet End** modifier point, whereas an addPageOverlay should ideally be called from an **On Page End** modifier point.

7.1.4 Manipulating Printer Profiles

This information applies to the Form Engine. Please see the chapter 11.1 Printer Profiles for a more thorough explanation of Printer Profiles.

To assist in manipulating Printer Profiles, Lasernet once again provides access to the page object.

7.1.4.1 The pages object

The *pages* object is used for add and clear printer profiles for each page. The pages are available in an array called pages, where the first page is named pages[1] and so on. There is one item for each page in the array. The example below shows one way of adding a printer profile to the first page:

```
pages[1].addPrinterProfile(”NameOfOutputModule”, ”NameOfProfile”, true);
```

The page object is referred to by the variable ‘pages’, which makes it possible to access the page array. The value, [1], signifies the first page is required, against which the command addPrinterProfile is appended. This method is used for instructing Lasernet to add a printer profile to that particular page. Inside the parentheses three parameters can be specified. The first parameter contains the name of the output module on which to add the profile. Note: this is the simple name of the module without any computer name prefixed. The second parameter tells Lasernet the name of the Printer Profile that you want to add for that module. This profile must have been created on the Printer Profiles page in the Lasernet Developer. The third parameter contains a Boolean (true/false) value where “true” means that Lasernet will remove any existing profiles for the specified module before adding the new profile, and false means that Lasernet will keep any existing profiles.

In general, existing profiles would normally need to be removed, since the output module will only use the first profile in the list. But to allow for backwards compatibility as well as future enhancements the parameter is still in use.

7.1.4.2 Shortcut functions

To make it easier to add a printer profile to a page, Lasernet provides two shortcut functions: addPrinterProfile and addPagePrinterProfile, to add a printer profile to all pages or to just the current page respectively.

```
addPrinterProfile(“ModuleName”, “ProfileName”);
addPagePrinterProfile(“ModuleName”, “ProfileName”);
```

It is best practice to execute the addPrinterProfile function from an **On Sheet End** modifier point, whereas addPagePrinterProfile is best practice to execute from an **On Page End** modifier point.

Note that both of the above functions will replace any existing printer profiles for a given module!

7.1.5 Manipulating Rearranges

This information applies to scripting in the Form Engine.

When creating rearranges in Lasernet, it is possible to manipulate the content and properties of the rearrange using a script. This is especially useful when calculating checksums for barcodes, etc.

7.1.5.1 CurrentRearrange

The CurrentRearrange variable is available for scripts being run from a named rearrange. The name should be unique otherwise it can cause confusion in Lasernet and the wrong rearrange could be assigned to the CurrentRearrange. It is very important that a rearrange has a name. If it does not, CurrentRearrange will not be assigned. Lasernet automatically assigns a name to the rearrange if **Script active** has been checked.

When the conditions above have been met, Lasernet assigns an InputRearrange to CurrentRearrange. This can then be used from a script to access the data which has been picked up from the left side. The data can then be manipulated and returned as the output value of the rearrange.

The script must return a value if the rearrange is to display anything. If you would like the rearrange to show empty, simply return an empty string. The script can do the calculation directly or call a function in one of the other script files.

```
'#' + CurrentRearrange.text + '#';
calcTheRearrange();
```

Besides manipulating the contents of a rearrange it is also possible to change the appearance of the rearrange. An example could be to display negative numbers in red.

```
if (CurrentRearrange.number < 0)
CurrentRearrange.color = red; // Constant defined in system.qs
```

You can even hide a rearrange by setting its color to white.

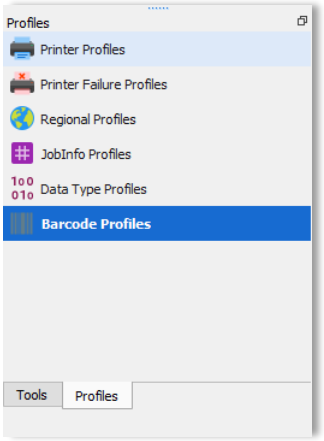
```
CurrentRearrange.color = white; // Constant defined in system.qs
```

8 Barcodes.

8.1 Barcode Profiles

A large range of built-in barcodes can be inserted into the output view of a text form. A barcode can be inserted and delivered as a print output or embedded into PDF or TIFF files for archiving, mailing etc.

A barcode profile is useful if the same barcode, with the same settings, is used in many different forms/labels. Changing settings for a profile, will affect any form where the barcode profile is inserted. This is a powerful way to manage the barcode globally and ensure that barcode settings are the same for all forms.

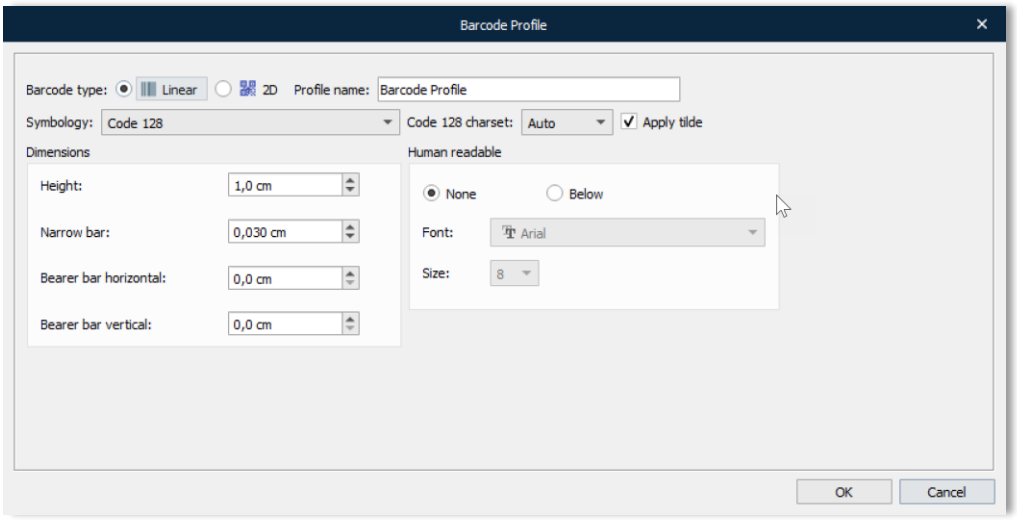


A range of linear and 2D barcodes are included in Lasernet as standard. The barcodes can be inserted as images in to forms designed via the Lasernet Form Editor, based on values in the incoming job data.

You are also able to insert barcodes into forms/labels without first creating a profile. Barcodes are created inside a form, but barcode settings cannot be inherited from other forms. Creating barcodes without profiles are convenient if only few forms have barcodes included.

8.2 Linear Barcodes

Select linear barcode as barcode type, the symbology (name of barcode) and type in a profile name describing which barcode you want to define.



Settings for dimensions check digit on/off and human readable specifications can be defined for the individual barcode profiles.

8.2.1 Dimensions - basic settings for linear barcodes

Most settings are common for linear barcodes.

Bar Height	1	The height of the barcode in centimetres (CM).
Narrow Bar Width	0.03	NarrowBarWidth is the width in centimetres of the narrow bars. This is also referred to as the X dimension. The default is 0.03 CM, which is about .012" or 12 mils. This value may need to be increased if the scanner being used cannot read barcodes with small X dimensions.
Bearer Bar Horizontal	0	The width of the horizontal bearer bars as a multiple of the XDimension; valid options are 0-10.
Bearer Bar Vertical	0	The width of the vertical bearer bars as a multiple of the XDimension; valid options are 0-10.

If the barcode has support for human readable characters, you can turn the setting on and define the font type and size you want to use.

8.2.2 Supported linear barcodes

More detailed information about linear barcodes can be found here:

http://www.idautomation.com/barcode-faq/

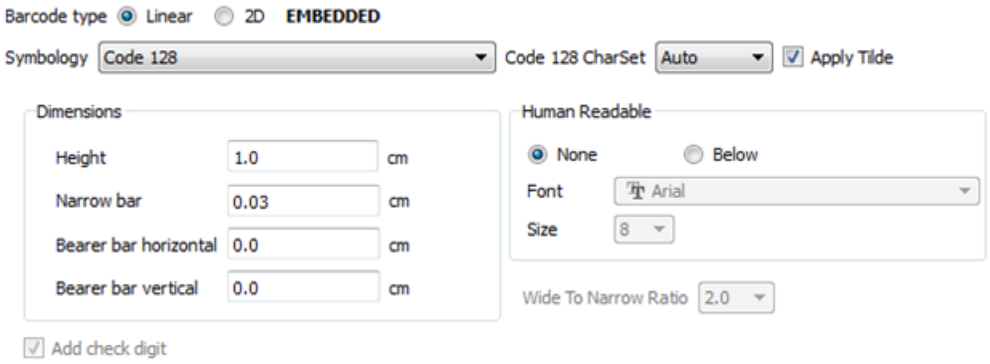
	**Code 128** - Alpha-numeric barcode with three character sets. Supports Code-128, GS1-128 (Formerly known as UCC/EAN-128) and ISBT-128.
	**Code 39 -** An alpha-numeric bar code that encodes uppercase letters, numbers and some symbols; it is also referred to as Barcode/39, the 3 of 9 Code and LOGMARS Code.
	**Code 93 -** Similar to Code 39, but requires two checksum characters.
	**Codabar -** A numeric barcode encoding numbers with a slightly higher density than Code 39.

	**Interleaved 2 of 5 -** The Interleaved 2 of 5 barcode symbology encodes numbers in pairs, similar to Code 128 set C.
	**POSTNET -** Used by US post offices for mail delivery and tracking.
	**UPC, EAN & GTIN** - This is one of the most common barcode types. It is used to encode the GTIN and also used to create JAN, ISBN and Bookland barcodes.

8.2.3 Code 128

Code 128 Barcode Fonts encode numbers, symbols, uppercase and lowercase text as well as functions such as returns and tabs. All Code 128 barcode fonts require a start character, checksum character, and a stop character to create readable barcodes.

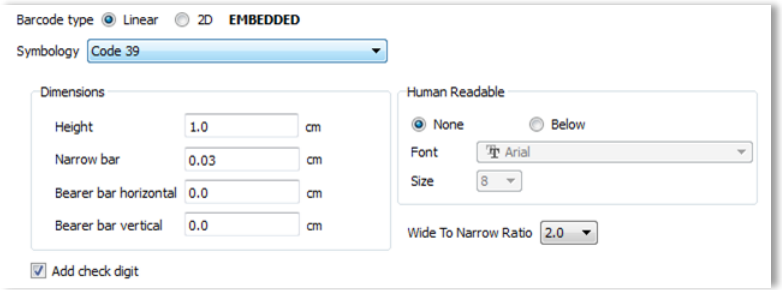
Set Linear as barcode type and Code 128 as symbology.



8.2.4 Code 39

Code 39 is a common barcode type used for various labels such as name badges, inventory and industrial applications. The symbology of the Code 39 character set consists of barcode symbols representing numbers 0-9, upper-case letters A-Z, the space character and the following symbols: - . $ / + %. Lower-case characters may also be easily encoded

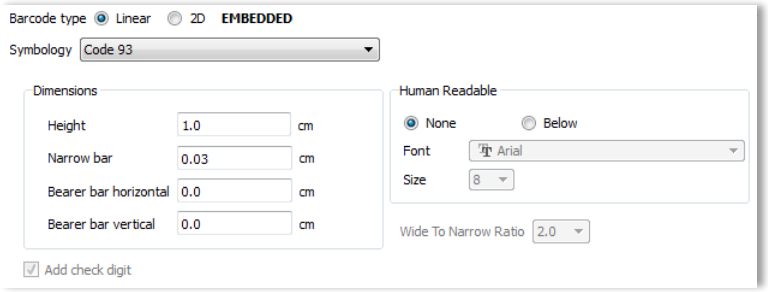
Set Linear as barcode type and Code 39 as symbology.



8.2.5 Code 93

Code 93 is a barcode symbology to provide a higher density and data security enhancement to Code 39. It is an alphanumeric, variable length symbology. To generate the Code 93 Barcode Font, the start and stop digits and 2 check characters must be included.

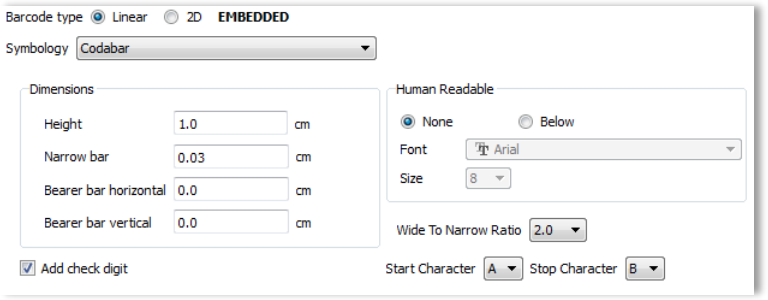
Set Linear as barcode type and Code 93 as symbology.



8.2.6 Codabar

Codabar is a low-density numeric bar code. It includes 16 characters: the numbers 0-9, plus "-", ".", ":", "$", "/", and "+". The use of four separate Start and Stop characters (A, B, C, and D) allows useful information to be encoded by characters normally considered as overhead. Can be of variable length and do not require a checksum.

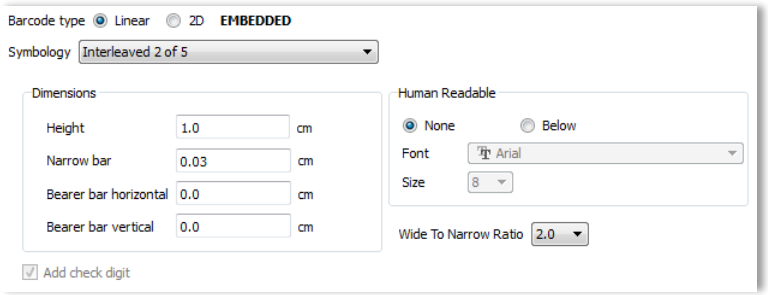
Set Linear as barcode type and Codabar as symbology.



8.2.7 Interleaved 2 of 5

Interleaved 2 of 5 is a numeric only barcode used to encode pairs of numbers into a self-checking, high-density barcode format. In this symbology, every two digits are interleaved with each other to create a single symbol. If a number string containing an odd number of digits needs to be encoded, a leading zero must be added to produce an even number of digits in the Interleaved 2 of 5 barcode.

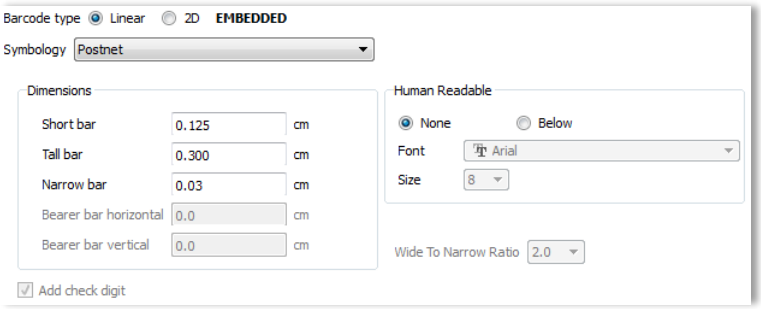
Set Linear as barcode type and Interleaved 2 of 5 as symbology.



8.2.8 Postnet

POSTNET (Postal Numeric Encoding Technique) is a barcode symbology used by the United States Postal Service to assist in directing mail.

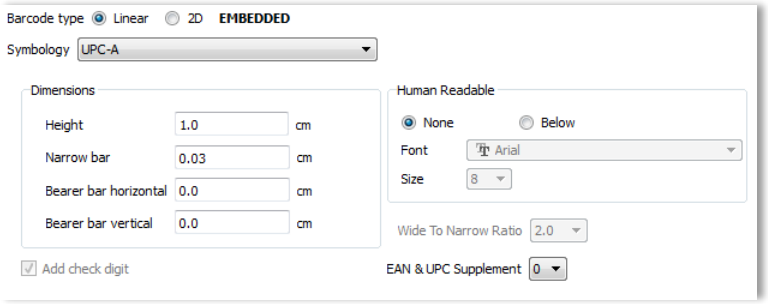
Set Linear as barcode type and Postnet as symbology.



8.2.9 UPC-A, EAN-8 and EAN-13

Along with the related EAN barcode, the UPC (Universal Product Code) is the barcode mainly used for scanning of trade items at the point of sale, per GS1 specifications.

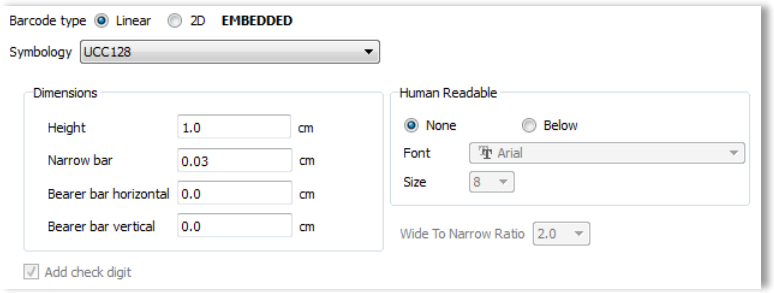
Set Linear as barcode type and UPC-A, EAN8 or EAN13 as symbology.



8.2.10 UCC128 (EAN-128, GS1-128)

GS1-128 is a standard of the GS1 implementation using the Code 128 barcode specification. The former correct name was UCC/EAN-128. Other no longer used names have included UCC-128 and EAN-128. GS1-128 uses a series of Application Identifiers to include additional data such as best before dates, batch numbers, quantities, weights and many other attributes needed by the user.

Set Linear as barcode type and UCC128 as symbology.

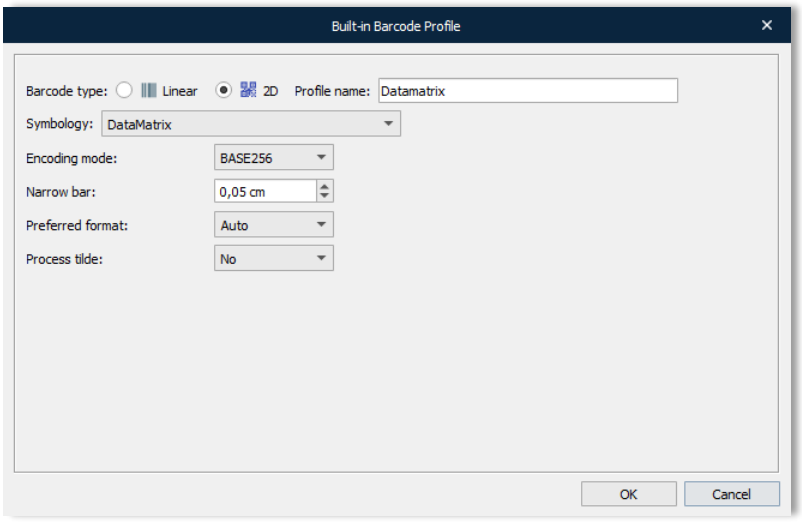


If you want read more about the supported linear barcodes, we recommend that you read the specifications here:

http://www.idautomation.com/barcode-faq/

8.3 2D Barcodes

Select 2D as barcode type, the symbology (name of barcode) and type in a profile name describing which barcode you want to define.



8.3.1 Supported 2D barcodes

More detailed information about 2D barcodes can be found here:

http://www.idautomation.com/barcode-faq/

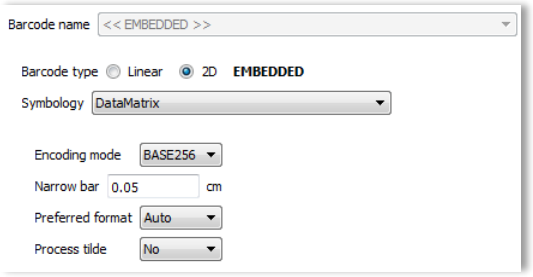
	**Data Matrix Fonts -** A matrix symbol that allows very efficient encoding of data into a square barcode with error correction.

	**MaxiCode Fonts -** Used primarily by UPS to route and track packages.
	**PDF417 Fonts -** This unique 2D barcode type is commonly used on FedEx shipping.
	**QR-Code Fonts -** A matrix symbol that is capable of encoding binary and ASCII characters. Asian/Kanji characters are not supported.

8.3.2 DataMatrix

DataMatrix code is a two-dimensional matrix barcode. The information to be encoded can be text or numeric data. Usual data size is from a few bytes up to 1556 bytes. The length of the encoded data depends on the number of cells in the matrix. Error correction codes are often used to increase reliability: even if one or more cells is damaged so it is unreadable, the message can still be read. A Data Matrix symbol can store up to 2,335 alphanumeric characters.

Set 2D as barcode type and Datamatrix as symbology.



8.3.2.1 Settings

Property Name	Default	Property Description
Encoding Mode	BASE256	Valid Encoding Mode values are ASCII, C40, TEXT or BASE256.
Narrow Bar	0.05	The width in centimetres of the narrow bars; which is also referred to as the X dimension. The default is 0.05 CM which is about .02" or 20 mils. This value may need to be increased if the scanner cannot read barcodes with small X dimensions. If using a high quality barcode imager, this value may be decreased to obtain a higher density barcode.
Preferred Format	-1	Sets the preferred Data Matrix formats represented by a number; valid values are from 0 (10X10) to 23 (144X144) and from 24 (8X18) to 29 (16X48); the default value of -1 is used for automatic formatting.
Process Tilde	No	When set to "Yes", the tilde (~) will be processed as explained in the DataMatrix Barcode FAQ.

8.3.2.2 DataMatrix Encoding Modes

By default, the encoding mode for most components is BASE256. If the choice is to encode text or numbers only and size is a concern, a change of the encoding mode to ASCII, TEXT or C40 may produce a smaller symbol. The data represented in the symbol may be compressed using one of the following modes:

- **ASCII** is used to encode data that mainly contains ASCII characters (0-127). It encodes approximately one alphanumeric or two numeric characters per byte. As a general rule, use ASCII to encode text that includes uppercase and lowercase letters with or without numbers and punctuation.
- **C40** is used to encode data that contains only numeric and upper-case characters. C40 encodes approximately three alphanumeric data characters into two bytes.
- **TEXT** is used to encode data that mainly contains numeric and lowercase characters. TEXT encodes approximately three alphanumeric data characters into two bytes.
- **BASE256** is used to encode images, double-byte characters, binary data and 8-bit values.

8.3.2.3 DataMatrix Formats

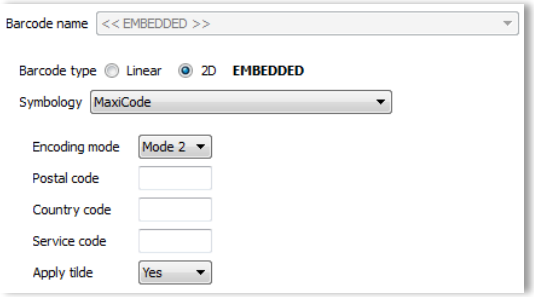
Data Matrix ECC 200 barcode products support all formats. The following table contains the size, capacity and error correction features of each format. By default, the encoding mode is **BASE256** (or binary) for compatibility reasons. The encoding mode may be changed to reduce the symbol size in some situations. The chart below illustrates the smallest symbol size, in the best-case scenario, for the amount of data encoded.

Format Number	Size	Max Numeric Capacity	Max Alphanumeric capacity	Max Binary capacity	Max Correctable Error/Erasure
0	10 x 10	6	3	1	2
1	12 x 12	10	6	3	3
2	14 x 14	16	10	6	5/7
3	16 x 16	24	16	10	6/9
4	18 x 18	36	25	16	7/11
5	20 x 20	44	31	20	9/15
6	22 x 22	60	43	28	10/17
7	24 x 24	72	52	34	12/21
8	26 x 26	88	64	42	14/25
9	32 x 32	124	91	60	18/33
10	36 x 36	172	127	84	21/39
11	40 x 40	228	169	112	24/45
12	44 x 44	288	214	142	28/53
13	48 x 48	348	259	172	34/65
14	52 x 52	408	304	202	42/78
15	64 x 64	560	418	278	56/106
16	72 x 72	736	550	366	72/132
17	80 x 80	912	682	454	96/180
18	88 x 88	1152	862	574	112/212
19	96 x 96	1392	1042	694	136/260
20	104 x 104	1632	1222	814	168/318
21	120 x 120	2100	1573	1048	204/390
22	132 x 132	2608	1954	1302	248/472
23	144 x 144	3116	2335	1556	310/590
24	8 x 18	10	6	3	3
25	8 x 32	20	13	8	5
26	12 x 26	32	22	14	7/11
27	12 x 36	44	31	20	9/15
28	16 x 36	64	46	30	12/21
29	16 x 48	98	72	47	14/25

8.3.3 MaxiCode

Maxicode is an international 2D (two-dimensional) barcode that is currently used by UPS on shipping labels for world-wide addressing and package sorting. MaxiCode symbols are fixed in size and are made up of offset rows of hexagonal modules arranged around a unique finder pattern. MaxiCode includes error correction, which enables the symbol to be decoded when it is slightly damaged.

Set 2D as barcode type and MaxiCode as symbology.



MaxiCode symbols encode two messages; a primary message and a secondary message. The primary message usually encodes the postal code, country code and the class of service number. The secondary message usually encodes address data, but can encode other types of information as well.

MaxiCode barcodes have different modes of operation.

- 2 = **US Carrier** with postal codes up to 9 digits in length.
  Approximately 93 characters may be encoded in this mode.
- 3 = **International Carrier** with alpha-numeric postal codes up to 6 digits in length.
  Approximately 93 characters may be encoded in this mode.
- 4 = **Standard Symbol** encodes general information for purposes other than the shipping industry.
  Approximately 90 characters may be encoded in this mode.
- 5 = **Secure Symbol** encodes general information with more error correction.
  Approximately 74 characters may be encoded in this mode.
- 6 = **Reader Program** allows scanner manufacturers to program barcode readers.

8.3.3.1 Symbol Size & Tolerance

The size and tolerance range of the MaxiCode symbol is noted below in millimetres. The sizes listed include the required quiet zone of one white hexagon on each side of the symbol, referred to as 1x. The allowable tolerance from the nominal size is approximately 5%.

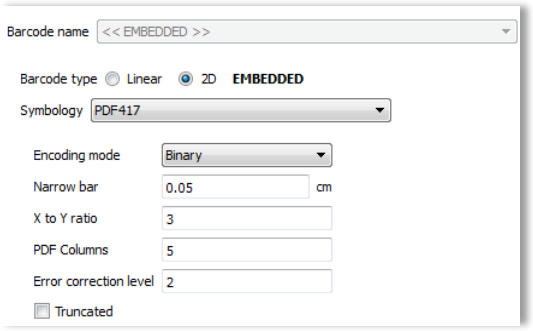
	Nominal	Minimum	Maximum	IDAutomation MaxiCode Font at 6 Points*
Width	28.14mm	26.48mm	29.79mm	28.10mm
Height	26.91mm	25.32mm	28.49mm	26.70mm
Bullseye	07.74mm	07.35mm	08.16mm	07.72mm

The printed size may vary slightly depending on the printer and labels used. We recommend printing the MaxiCode Font at 300 DPI or greater.

8.3.4 PDF417

PDF417 is a multi-row, variable-length symbology with high data capacity and error-correction capability. PDF417 has some unique features which make it the most widely used 2D symbology. A PDF417 symbol can be read by linear scanners, laser scanners or two-dimensional scanners. PDF417 is capable of encoding more than 1100 bytes, 1800 text characters or 2710 digits.

Set 2D as barcode type and PDF417 as symbology.



Property Name	Default	Property Description
Error Correction Level	2	The Reed Solomon error correction level encoded in the symbol.
PDF Columns	5	The number of data columns in the PDF417 barcode. The default is 5 and the maximum is 30.
Encoding Mode	Binary	The mode of compaction used to encode data in the symbol. When set to "Text," a smaller symbol may be created. Text mode encodes all characters on the US keyboard plus returns and tabs. Binary mode encodes ASCII 1 to 254.
PDFRows	0	Sets the number of rows. It is recommended to leave this setting at 0, the default.
Truncated	0	A truncated PDF417 symbol is more area efficient than normal PDF417. When set to true, the right hand side of the PDF417 is removed or truncated.
XtoYRatio	3	The X multiple height of individual cells; default = 3, usually 2 to 4 times the NarrowBarCM (X).

Large amounts of text and data can be stored securely and inexpensively when using the PDF417 barcode symbology. The printed symbol consists of several linear rows of stacked codewords. Each codeword represents 1 of 929 possible values from one of three different clusters. A different cluster is chosen for each row, repeating after every three rows. Because the codewords in each cluster are unique, the scanner is able to determine what line each cluster is from.

8.3.4.1 PDF417 Error Correction Levels

PDF417 uses Reed Solomon error correction instead of check digits. This error correction allows the symbol to endure some damage without causing loss of data. AIM standards recommend a minimum error correction level of 2. The error correction level depends on the amount of data that needs to be encoded, the size and the amount of symbol damage that could occur. The error correction levels range from 0 to 8. The chart below indicates the number of error correction codewords that are added to the printed symbol and AIM recommendations for the EC level.

Fig. 1: AIM recommended EC levels

EC Level	0	1	2	3	4	5	6	7	8
EC Codewords Generated	2	4	6	8	16	32	64	128	512
Data Codewords			1-40	41-160	161-320	321-863			
Data Bytes Encoded			1-56	57-192	193-384	385-1035			

8.3.4.2 X and Y Dimensions

The X dimension is the width of the narrowest bar in a printed codeword. The Y dimension is the height of each row within the PDF417 symbol. The PDF417 barcode is usually printed at an X to Y ratio of 1:2 to 1:5, with 1:3 being the most popular. By lowering the ratio, a significant amount of space can be saved; however, some scanners cannot read X to Y ratios of less than 1:3. Most scanners read PDF417 barcodes well at 1:3. When creating symbols with more than 10 columns we recommend using a ratio of 1:4 or 1:5.

8.3.4.3 Row and Column Limits

The number of rows and columns can be selected, allowing the symbol to be created in various forms. However, the PDF417 barcode symbol is limited to 30 columns and 90 rows. Keep in mind when selecting columns, only the number of data columns in the symbol are being selected. The normal PDF417 barcode symbol has two row start columns and two row stop columns. Truncated PDF417 contains only two row start columns. Some scanners and decoders cannot dependably read over 20 columns.

8.3.4.4 Truncated PDF417

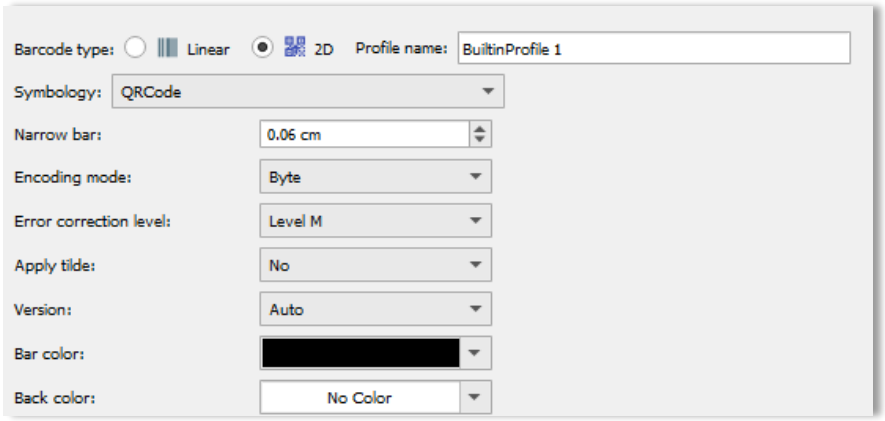
A truncated PDF417 symbol uses less area than the normal PDF417 barcode. By selecting this option, the right-hand side of the symbol is removed or truncated. This option should be used primarily in a clean environment, since it is more susceptible to damage.

8.3.5 QR Code

QR Code is a very efficient, two-dimensional (2D) barcode symbology that uses a small area of square modules with a unique perimeter pattern, which helps the barcode scanner determine cell locations and decode the QR Code symbol. Characters, numbers, text and actual bytes of data may be encoded, including Unicode characters and images. Implementation of QR Code is based on the ISO/IEC 18004:2006 standard. Asian/Kanji characters are not supported.

Note: Colored QR codes are supported, in addition to inverted QR codes – for example, white on a black background.

Set Barcode type to 2D and Symbology to QRCode.



Property Name	Default	Property Description
ProcessTilde	False	When set to "True", the format ~d??? may be used to specify the ASCII code of the character to be encoded. Commonly used ASCII codes are ~d009 which is a tab function and ~d013 which is a return function.
EncodingMode	Byte	The mode of compaction used to encode data in the symbol. Valid values are E_BYTE, E_ALPHANUMERIC, and E_NUMERIC mode.
Version	AUTO	Sets the size of the symbol; valid values are from 1 (21X21) to 40 (177X177); the default value of "AUTO" is used for automatic formatting.
ErrorCorrection	M	The error correction level encoded in the symbol. Valid values are ECL_L, ECL_M, ECL_Q, ECL_H. Higher error correction creates a larger symbol that can withstand more damage.
XDimensionCM	0.06	The width in centimetres of the narrow bars. This value may need to be increased if the scanner cannot read barcodes with small X dimensions. When working with a high quality imager, this value may be decreased to create a smaller symbol.

8.3.5.1 Error Correction and Encoding Modes

By default, the encoding mode for most components is "byte" and an error correction level of M. If you are only encoding text and size is a concern, changing the encoding mode to Alpha-Numeric may produce a smaller symbol. The data represented in the symbol may be encoded using one of the following modes:

Encoding Mode	Parameter Selection*	Description
Byte	0	Encodes lower case letters, text, images, double-byte characters, binary data and 8 bit values.
Alpha-Numeric	1	Encodes only numbers and uppercase letters. In this encoding mode, lower case letters will be converted to upper case. All other data will be filtered out.
Numeric	2	Encodes only numbers; all other data will be ignored.

Four levels of Reed-Solomon error correction are referred to as L, M, Q and H. The error correction levels allow verification of data and recovery in the event that part of the symbol is damaged. Increasing the error correction level increases the symbol size and reduces data capacity. The percentage of recovery and capacity noted below are approximate.

Error Correction Level	% of Recovery	Byte Capacity at Size 24	Parameter Selection
L	7%	1171	2
M	15%	911	0
Q	25%	661	3
H	30%	511	1

8.3.5.2 Symbol Version

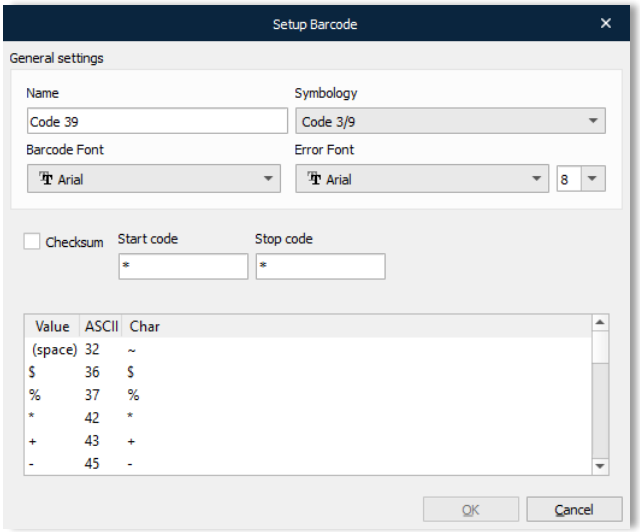
The version is the size of the symbol from (1) 21x21 to (40) 177x177. Zero is the default selection. If the symbol needs to be larger than the selection, the component automatically overrides this value.

If you want read more about the supported 2D barcodes we recommend reading the specifications from:

http://www.idautomation.com/barcode-faq/

8.4 Linear Barcodes (Symbology and TrueType Fonts)

Inserting barcodes via TrueType fonts is supported and compatible with barcode fonts from www.elfring.com, which must be bought and installed separately. Lasernet is able to calculate the symbology for a list of linear, insert stop, start code and check sums. The symbology for a TrueType font is maintained via profiles only and cannot be added in the settings of a barcode rearrange like the built-in barcodes.



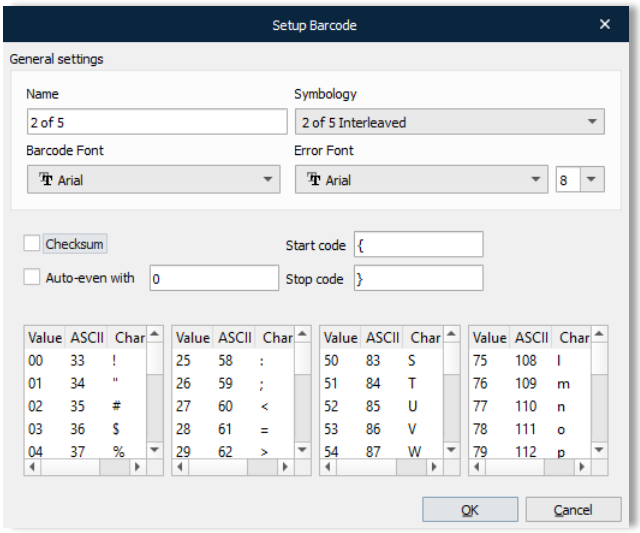
Generic barcode support is provided in Lasernet for the following symbologies:

- Code 128 and EAN 128
- Code 2 of 5 Interleaved
- Code 3/9
- EAN 8/13

A barcode symbology defines what characters the barcode can represent, how to calculate checksums (if any) and other properties specific for the individual barcode.

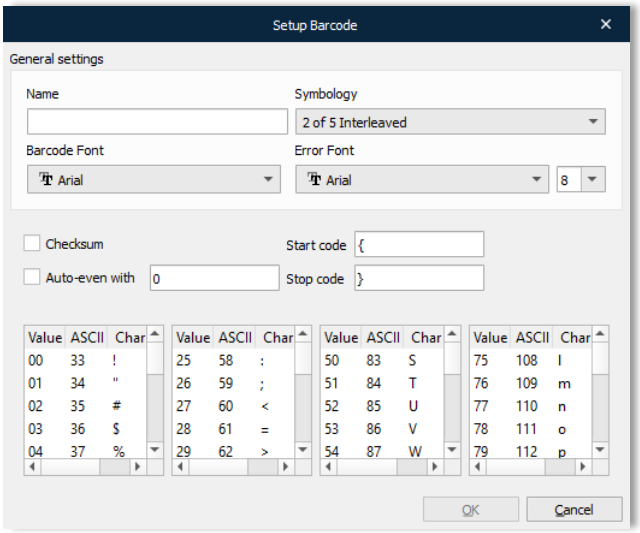
For Lasernet to produce its own barcodes, TrueType barcode fonts are required. Lasernet uses the Elfring TrueType fonts.

To create a new barcode click **Add** then choose which symbology to use in the dialog pop-up window.



Name of template	The name of the barcode - must be unique. Lasernet will not allow you to save the barcode if another barcode with the same name already exists.
Barcode font	What font will be used for encoding the barcode.
Error font (Underneath the Symbology dropdown list).	Sometimes an encoding error will occur in the barcode. This sets what font to use for printing this error.
Font size	What font size must be used for error fonts.
Symbology	When you change the symbology the settings for each type appear under general settings:

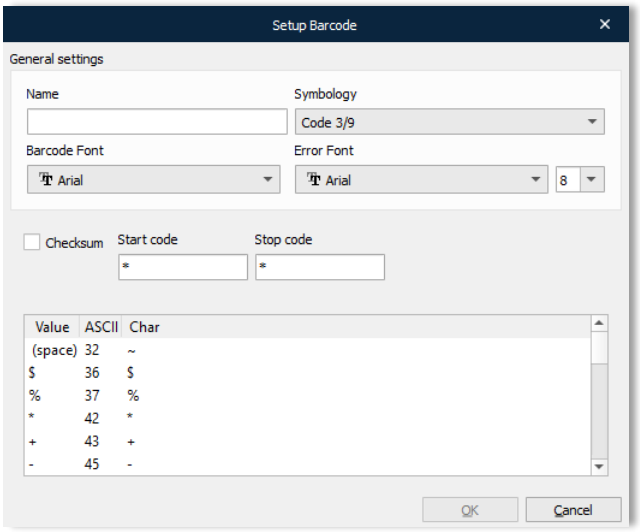
8.4.1 2 of 5 Interleaved



2 of 5 Interleaved encodes digits, lowercase and uppercase characters and a few special characters. The Character mapping tables are only for information. They should not be edited. At present Lasernet only supports Elfring barcode fonts.

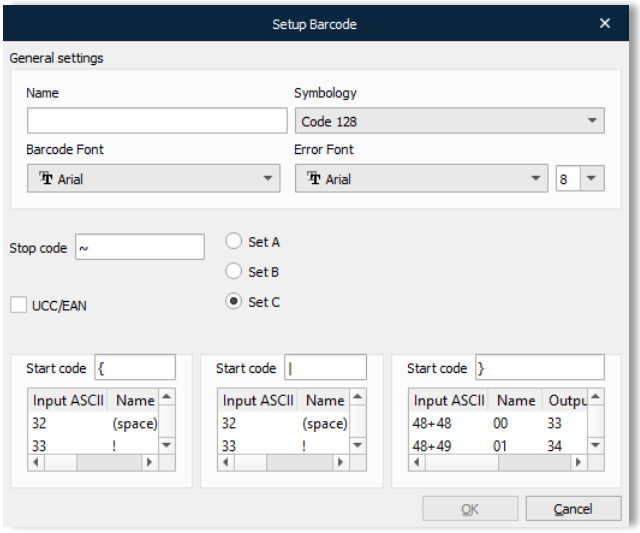
This symbology requires that the resulting barcode encodes an even number of characters. This includes the checksum digit. The **Auto-even with** property is used to ensure this. If this is not checked and the resulting barcode does not have an even number of characters, an error is printed instead.

8.4.2 Code 3/9



A simplistic symbology - encodes capital letters, digits and few special characters. It provides an option for a checksum. Please note that you need an intelligent reader to actually verify this checksum, since it is just a digit extra in the barcode.

8.4.3 Code 128



Code 128 is one of the most advanced barcodes, capable of encoding nearly all characters in the standard ASCII character set. It is also able to encode pure digits very efficiently.

The three tables show how the symbology encodes all these characters, using one of three subsets (A, B and C). Subset A focuses on capital letters, single digits, special characters and ASCII control characters. Subset B focuses on both lowercase and uppercase characters, single digits and a few special characters. Subset C concentrates on the expression of two digits for each barcode digit. This means that if you only need to encode digits you can use subset C to limit the width of the barcode.

For each subset there is a conversion table which consists of four columns:

- Input ASCII
- Name
- Output ASCII
- Char

Input ASCII shows the ASCII values that you need to write to get a specific barcode character. Name is the name that the barcode symbology gives that character.

Output ASCII shows the ASCII value of the character in the barcode font. Char is just a character representation of that value.

Three of four columns are editable: Input ASCII, Output ASCII and Char (Output ASCII and Char automatically follow one another when changed).

Reasons for changing the Input ASCII may occur if it was not possible to generate a specific character in your output system.

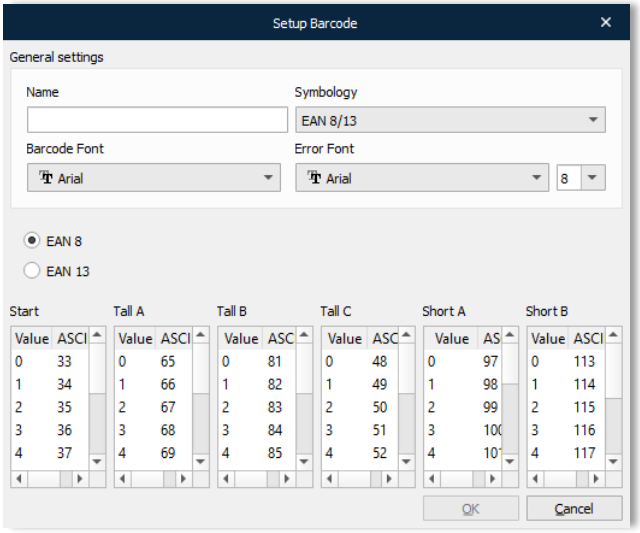
Output ASCII/Char values may need to be changed if another barcode font is chosen over Elfring, as alternate character mapping may be used by the new font.

Subset C uses Input ASCII combined with a '+', for example, 53+50. This means that you should output *52* to Lasernet in order to get the barcode to encode those digits. Please note that only Subset C supports the '+' syntax.

Code 128 or EAN 128 supports subset changing during the barcode.

This means Lasernet can create a Code 128 A, Code 128 B and Code 128 C. Rather than two subsets in a single barcode e.g., Code 128 B+C, or Code 128 A+B. The same rules apply for EAN 128.

8.4.4 EAN 8/13



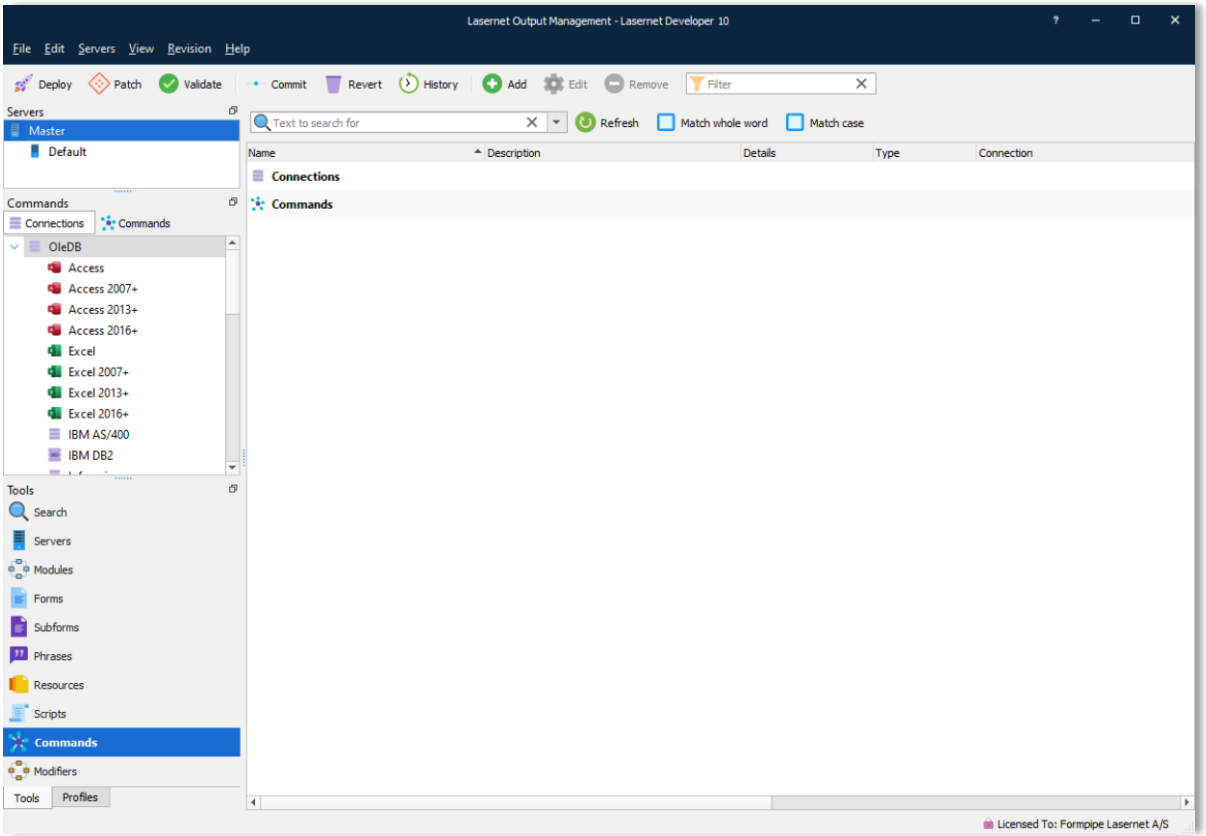
This barcode is widely used for food packaging. It encodes 8 or 13 digits. The mapping table is subject to the same limitations as for Code 3/9.

9 Databases.

9.1 Database Connections

In order to work with a database a connection is required. The parameters required for each type of database vary, therefore a general overview is provided here, together with more specific instructions for commonly used databases.

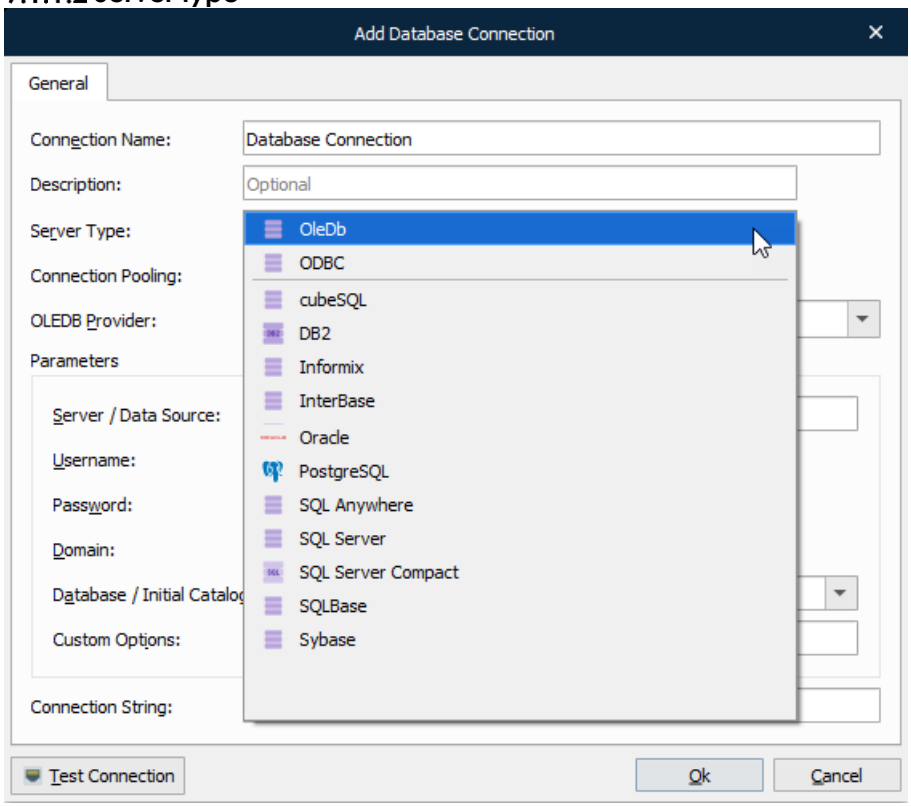
Note: This area also contains connections that are used to access Microsoft Azure services, such as SharePoint (see section 9.1.5 SharePoint) and Azure Storage (see section 9.1.6 Azure Storage).



9.1.1.1 Connection name

A unique name used for referencing the connection in the commands.

9.1.1.2 Server type



9.1.2 OleDb

All major databases use OleDb drivers. These are supported via the OleDb framework which provides access to almost every function of the database.

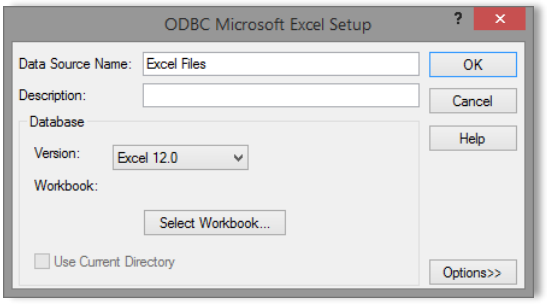
The necessary drivers for your database must be installed on the computer running Lasernet. They also need to be installed on the computer that runs the Lasernet Developer otherwise connections and commands cannot be tested.

If Lasernet can't communicate via the OLEDB provider it will result in an error. This can typically happen if the connection string hasn't been defined as intended or Lasernet is not sharing the same 32- or 64-bit architecture as the OLEDB provider. In such scenario the error message, *"COM error when communicating with OleDb: Provider can't be found. It may not be properly installed",* will occur.

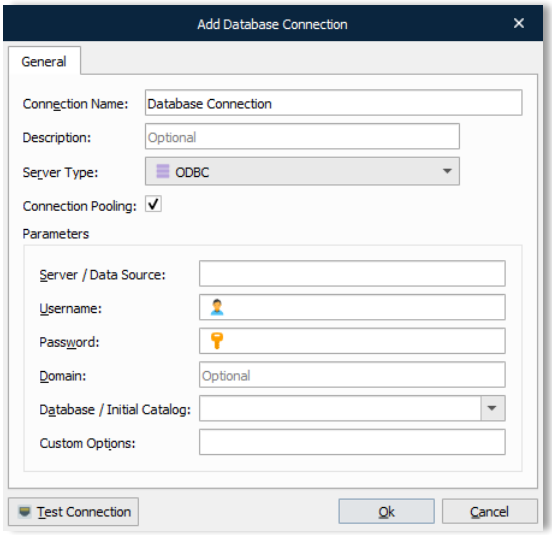
9.1.3 ODBC

To use ODBC an external client and driver components must be installed on the computer running Lasernet. Any ODBC client can access a DBMS (Database Management System) for which there is an ODBC driver. DBMS server is a back-end system, for example SQL Server, Oracle and AS/400, or any DBMS for which an ODBC driver exists.

ODBC drivers must be installed and maintained via the Windows ODBC Data Source Administration tool and support the same 32- or 64-bit architecture as Lasernet.

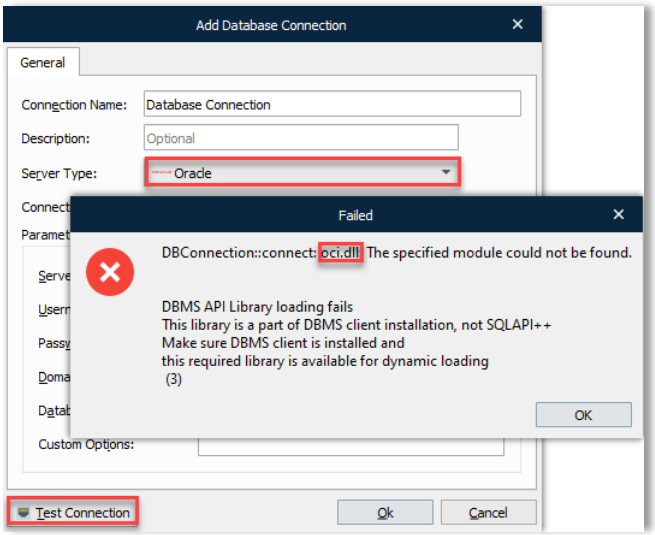


Info: When adding a Database Connection using ODBC in Lasernet Developer, the Data Source Name (located in the Windows ODBC Administration tool) has to be entered in the field for the Database / Initial Catalog value.



9.1.4 Native (SQLAPI)

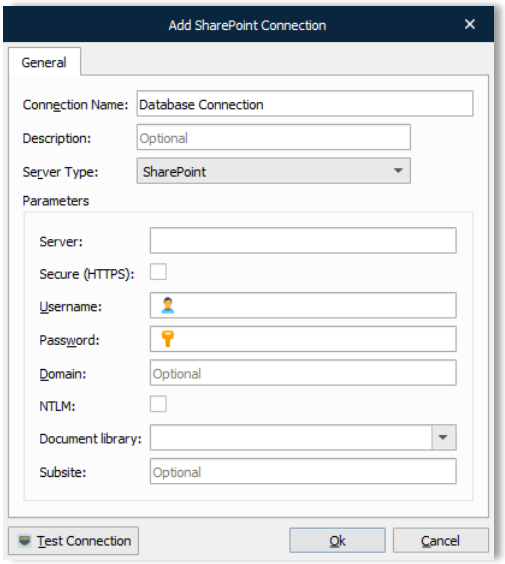
Lasernet uses a SQLAPI and Microsoft's OleDb framework as libraries. SQLAPI provides specific drivers for selected databases. These drivers are generally fast, and often require 3rd party drivers to be able to connect to the selected database.



Select one of the supported databases and click **Test Connection**. If the required driver is not available the connection will fail. The dialog will provide you with information about required 3rd party drivers to successfully connect to the selected database type. Ensure that the driver from the database provider has the same bit architecture as your Lasernet Developer and Lasernet Service and that it is available for both applications, if they are not installed on the same computer. The driver must be installed in the same directory as your Lasernet software.

9.1.5 SharePoint

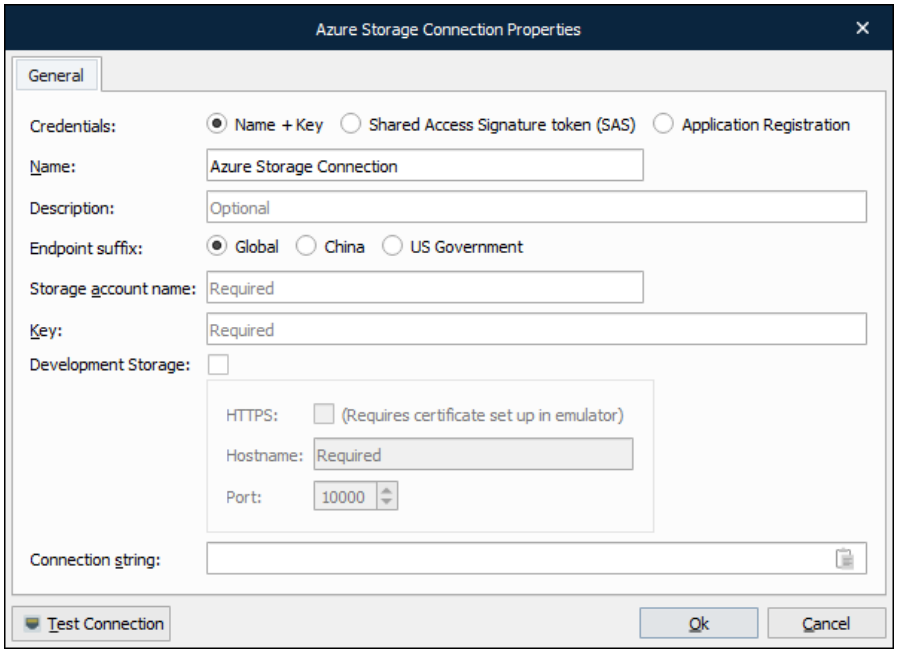
This connection is used for connecting to a SharePoint server.



Refer to the “Lasernet 10 – Microsoft SharePoint” manual for more information.

9.1.6 Azure Storage

This connection is used for connecting to Azure Storage.



Refer to the “Lasernet 10 – Azure” manual for more information.

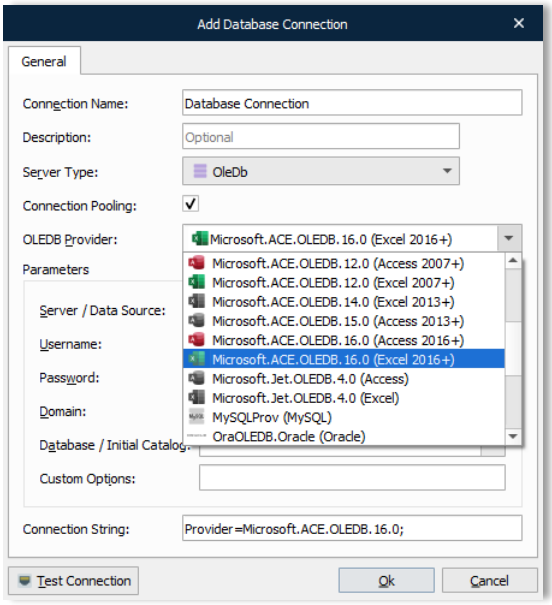
9.1.7 General settings

9.1.7.1 Connection Pooling

The Connection pooling option is activated by default when adding a new Database Connection. It works as a cache of database connections which can be reused when future requests to the database are required. Connection pools are used to enhance performance when executing commands on a database. Opening and maintaining a database connection for each user is costly and wastes resources. When using connection pooling, the connection is placed in the pool after it has been created and used over and over again, so that a new connection does not have to be established. If all the connections are being used, a new connection is made and is added to the pool. Connection pooling also cuts down on the amount of time a user must wait to establish a connection to the database.

When exporting or migrating a Database Connection from an earlier version that does not support pooling, the setting will not be activated automatically. Instead, Lasernet will connect to the database and behave as in older versions.

9.1.7.2 OleDb Provider



The OleDb Provider list is enabled when the server type is set to OleDb. There are many providers listed here however the list is not exhaustive. You can enter any provider you want to. The following web site, http://www.connectionstrings.com/, may help with specific details for your provider.

9.1.7.3 Server/Data source

This is usually the name of the server on which the database server runs. It corresponds to the OleDb Data Source setting. For Microsoft Access databases this is the full path to the .mdb file.

9.1.7.4 Username and Password

This is the username and the password for the connection. If using integrated security for OleDb and SQL Server they can be left empty and Integrated Security=SSPI can be set in **Custom Options**.

9.1.7.5 Database/Initial Catalog

This is the name of the database on the server. It corresponds to the OleDb **Initial Catalog** setting. For ODBC it is the name of the DSN (Data Source Name).

9.1.7.6 Custom Options

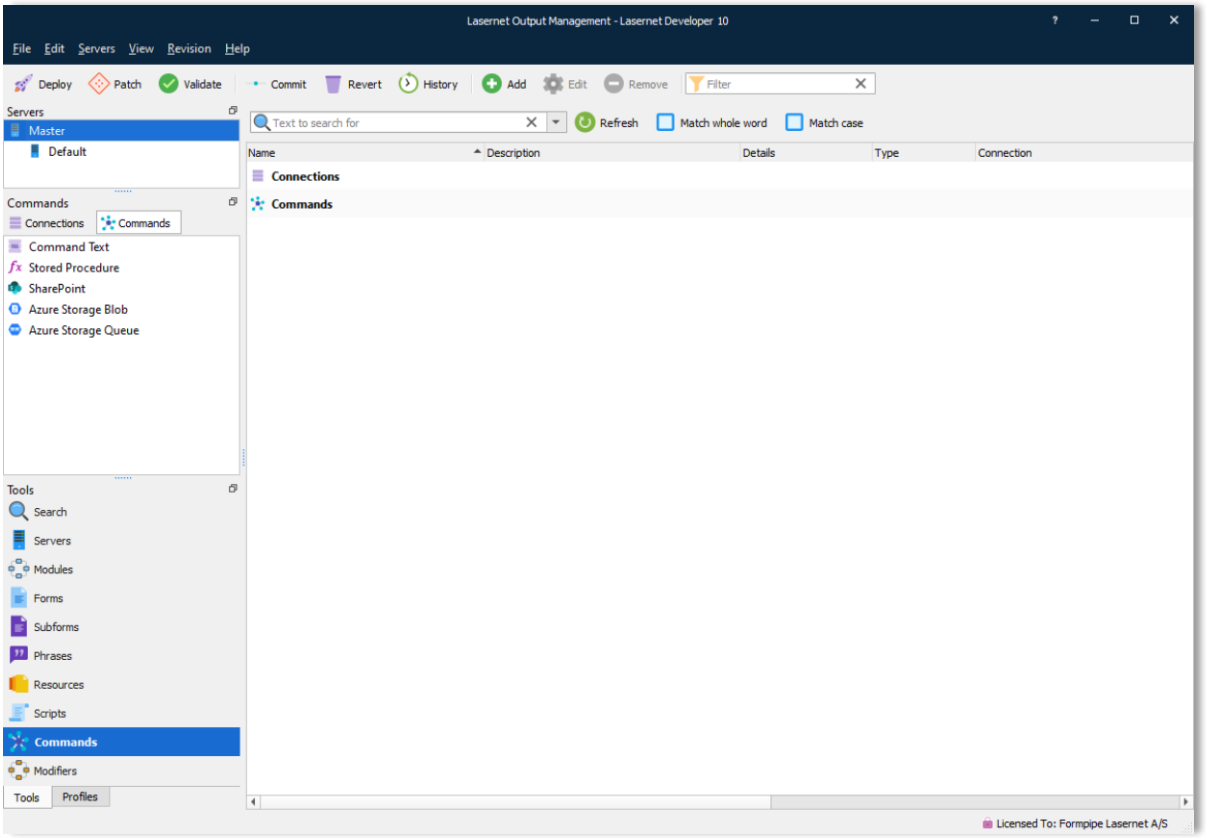
Custom Options are only used for OleDb and give the opportunity to set any specific options needed. One setting could be Integrated Security=SSPI for SQL Server.

9.1.7.7 Connection String

The Connection String field is visible/readonly when a specific OleDb provider is selected. In the case of an unsuccessful connection to a database server, you can set the connection string to **Custom (User defined)** and edit the Connection String manually.

9.2 Database Commands

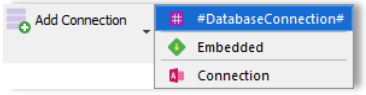
Database Commands are used for executing SQL queries against databases.



9.2.1 Command name

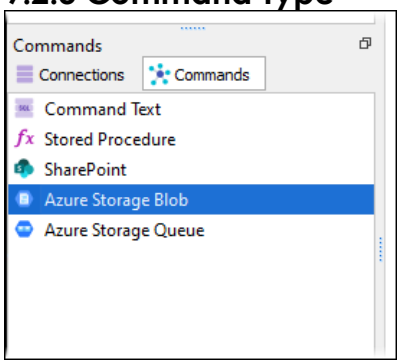
Command name is a unique name which is used for referencing the Database Command when calling it from either a modifier point or script.

9.2.2 Connection



Choose the connection to execute the command through. There is a special connection in the connection list called #DatabaseConnection#, which instructs Lasernet to use the connection name that is put in the JobInfo DatabaseConnection. This can be used for choosing the connection dynamically at runtime.

9.2.3 Command Type



Once the server type for the connection is defined, you can choose the type for the Command Text. The available commands are: Command Text, Stored Procedure, SharePoint, Azure Storage Blob, and Azure Storage Queue. Command Text and Store Procedure are used for Database Connections: OleDB, ODBC and Native. A SharePoint command is always used with a SharePoint Connection.

9.2.4 Command Text

This is the SQL query which is sent to the database server. The SQL query usually retrieves data from the database or updates/inserts/deletes data in the database. The syntax supported in the command text depends on the specific database backend used, but there are some common features which Lasernet makes available. One is substituting #JobInfo# markers with the content of the given JobInfo – e.g.

```
select * from customers where id = #CustomerID#
```

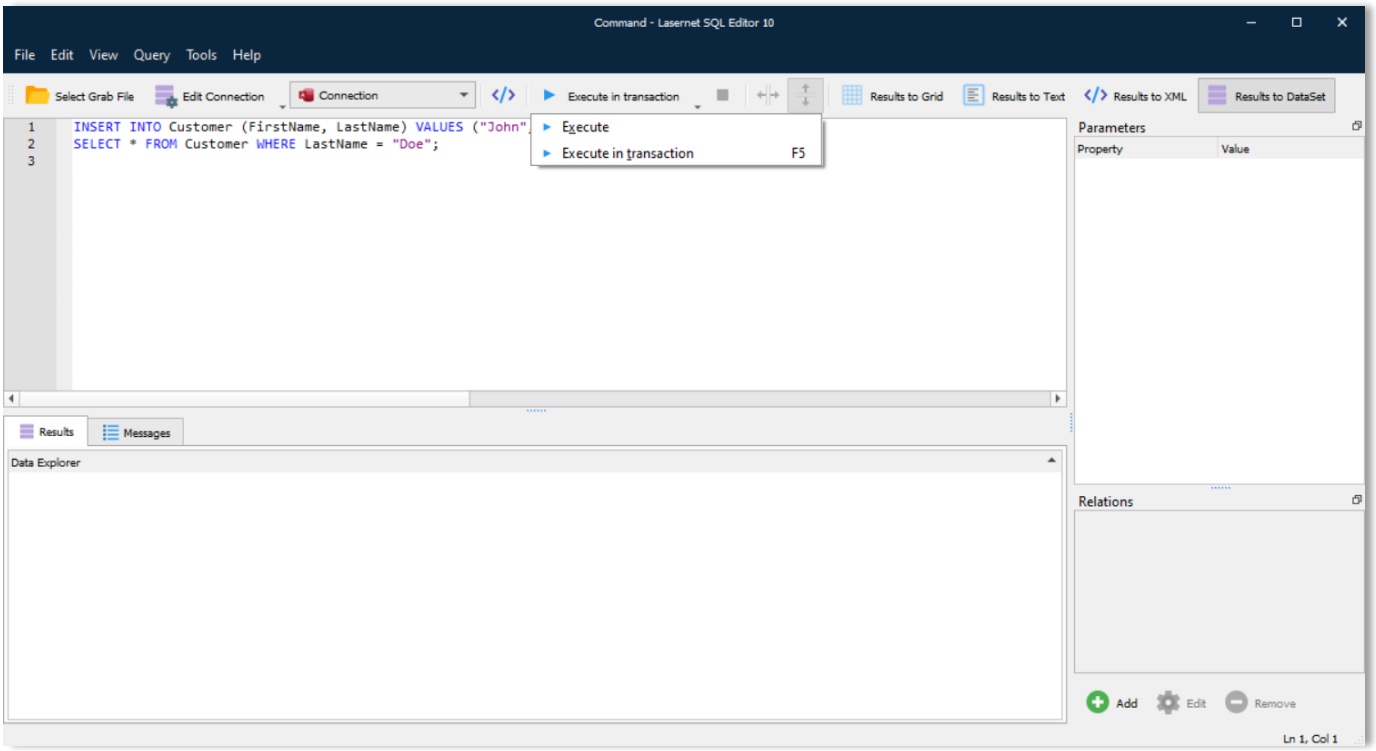
This is substituted with the value of the JobInfo CustomerID when the command is run. When simple numeric values are needed, this is the preferred way to customize your SELECT statement. But when using string values, this can cause trouble. Consider this:

```
select * from mytable where text = '#SomeText#'
```

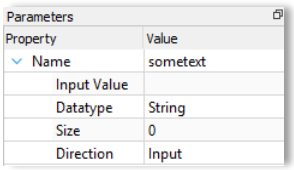
As before, Lasernet will substitute the #SomeText# with the actual value of that JobInfo. However, if a JobInfo contains text with an apostrophe it will not substitute correctly. Therefore binding must be used (recommended method). The example below shows the correct syntax:

```
select * from mytable where text = :sometext
```

Formatting the select statement this way makes '*:sometext'* a *bindable* parameter. In the *Parameters* list, the default values, data types etc, for the parameter can now be set:


Command - Lasernet SQL Editor 10
File Edit View Query Tools Help
Select Grab File
Edit Connection
Connection
Execute in transaction
Results to Grid
Results to Text
Results to XML
Results to DataSet
1 INSERT INTO Customer (FirstName, LastName) VALUES ("John"
2 SELECT * FROM Customer WHERE LastName = "Doe";
3
Execute
Execute in transaction F5
Parameters
Property
Value
Results
Messages
Data Explorer
Relations
Add
Edit
Remove
Ln 1, Col 1

When editing each parameter there are some options:



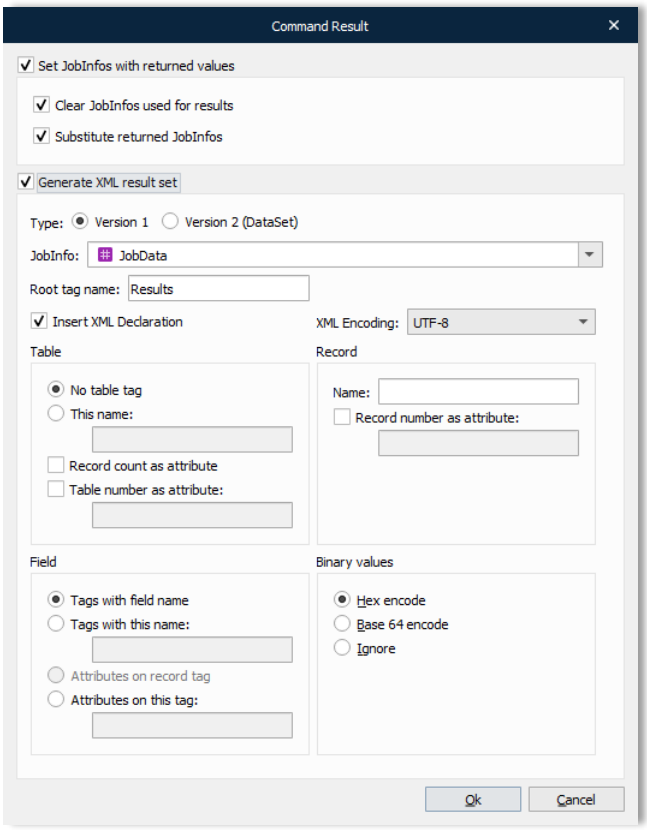
The Input Value field is used for setting the value of the parameter. This can be any #JobInfo# references which are automatically substituted.

The Data type field is used to set the type of the parameter. Usually, the database is quite strict about this setting however some databases are capable of converting types by themselves. Generally, types like String, Double, Long (an integer) and Long binary are used. Lasernet also uses the type to determine how to convert input and output values.

The Direction is used for telling both Lasernet and the database in which direction the data from the parameter flows. Input means that the data flows from Lasernet to the database. Output, which is Stored Procedure specific, means that the database backend fills the parameter with some data. Input/Output specifies that data flows in both directions.

9.2.5 Command result

In the **Tools → Command** Result setting you can specify the behaviour of return values for JobInfos or activate that the result should be generated as XML.



Set JobInfos with returned values
Checking this will cause Lasernet to return data from the database into JobInfos, which are named after the column names in the table of the database. This means that if you retrieve more than one row of data from the database, you will get arrays of JobInfos containing data.

It is possible to find out if anything was returned. This is done by checking the JobInfo RecordCount. If the value is 0 (zero), no records were returned. Output parameters always return a value.

Clear JobInfos used for results
This makes Lasernet automatically clear the JobInfos before starting to enter data into them.

Substitute returned JobInfos
When this is checked, Lasernet will run JobInfo substitution on the returned values before storing them in JobInfos. See chapter about JobInfo Substitution for more information.

Generate XML result set
Default data is retrieved as JobInfos from the database. It is also possible to return the data into an XML document. This is useful if data needs to be placed into a job to be rearranged in the Form Engine or the result stored into a file.

Activate this setting if you want to retrieve your database command result as an XML document. JobInfos containing field names and values from the database tables will also be created.

JobInfo
The name of the JobInfo that will contain the XML document.

By setting the value to JobData, the primary job parsing through Lasernet will now consist of the XML result set.

Root tag name
The root tag name of the XML document.

XML Encoding
The XML encoding to be used in the XML document.

Table tag name
Which (if any) tag name to be used for the table.

If *No table tag* is chosen, the attribute is placed on the root tag.

Record count as attribute sets the value of the RecordCount attribute to the number of records returned from the database.

Record tag name
For each record returned from the database this tag is used to enclose the records.

Record number as attribute sets the number of the record (starting from zero) in the attribute given.

Field tag name
Choose how to store the value in the XML document for each field in each record.

Tags with field name adds a tag for each field with the tag name set to the field name.

Tags with this name adds a tag with the given name for each field.

Attributes on record tag adds the field values as attributes on the record tag. Each attribute has the name from the field.

Attributes on this tag adds one tag with the given name and sets the values as attributes on that tag.

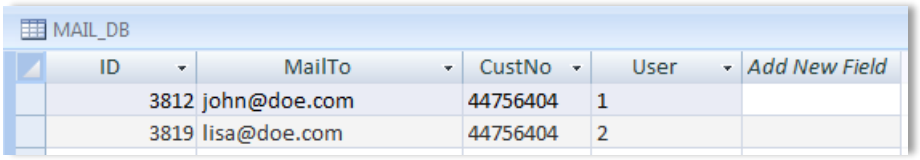
Binary values
Choose how to handle binary values.

- *Hex encode* encodes the value in this format
- *Base 64 encode* encodes the value in this format
- *Ignore* ignores binary values

Example

In this example all rows from the table MAIL_DB are selected where the field CustNo is equal to 44756404.

select * from MAIL_DB where CustNo = '44756404'



MAIL_DB

ID	MailTo	CustNo	User	Add New Field
3812	john@doe.com	44756404	1	
3819	lisa@doe.com	44756404	2	

In the Database Command Modifier “Generate XML result set” is activated and the parameters are defined as follows:

The result is an XML document stored into the JobInfo DBData with the following structure.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <LaserNet RecordCount="2">
  - <Row ID="0">
      <ID>3812</ID>
      <MailTo>john@doe.com</MailTo>
      <CustNo>44756404</CustNo>
      <User>1</User>
    </Row>
  - <Row ID="1">
      <ID>3819</ID>
      <MailTo>lisa@doe.com</MailTo>
      <CustNo>44756404</CustNo>
      <User>2</User>
    </Row>
  </LaserNet>
```

XML encoding is set to UTF-8.

Root tag named <LaserNet> with record count defined as an attribute (two records are selected).

Each record includes <Row> with ID defined as an attribute containing the record number.

Tags selected from the table are named with field names and the value of the field is added.

Other variants of the XML document can be defined by setting the appropriate properties.

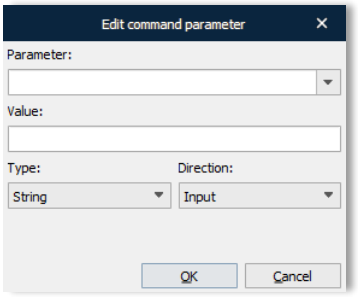
9.2.6 DataTable relations between queries in Database Command

You can specify how DataTables are related, making it possible to have master/detail relationship. This enables you to mail merge or create reports, for example, with customers and their orders, based on your query.

9.3 Stored Procedures

Calling a stored procedure is as easy as picking a database connection, selecting which procedure to call and filling in the parameters accordingly.

Lasernet supports input and output parameters as well as a return value from the procedure (OleDB connections only). Features may vary depending on the backend used. For example, Lasernet reads the possible parameters from some backends (SQL Server, Oracle etc.), while for others it's up to the user to add parameters manually via the Add/Remove buttons in the toolbar.



Parameter values support JobInfo substitution.

Parameters can have different directions; Input, Output, Input & Output and Return.

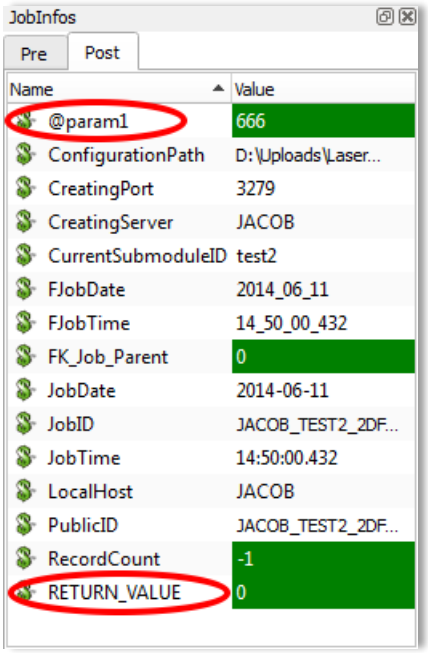
Input: Is sent to the backend.

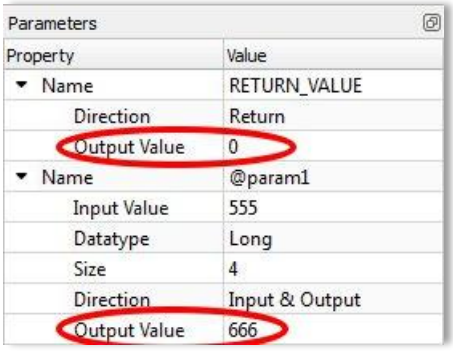
Output: Parameter is filled with value set in backend.

Input & Output: Is sent to the backend and filled with value set in the backend.

Return: A procedure can return an integer value called a return code to indicate the execution status of a procedure.

The returned values are inserted into JobInfos with the name of the parameter. The return value is sent in "RETURN_VALUE" JobInfo. For Example:

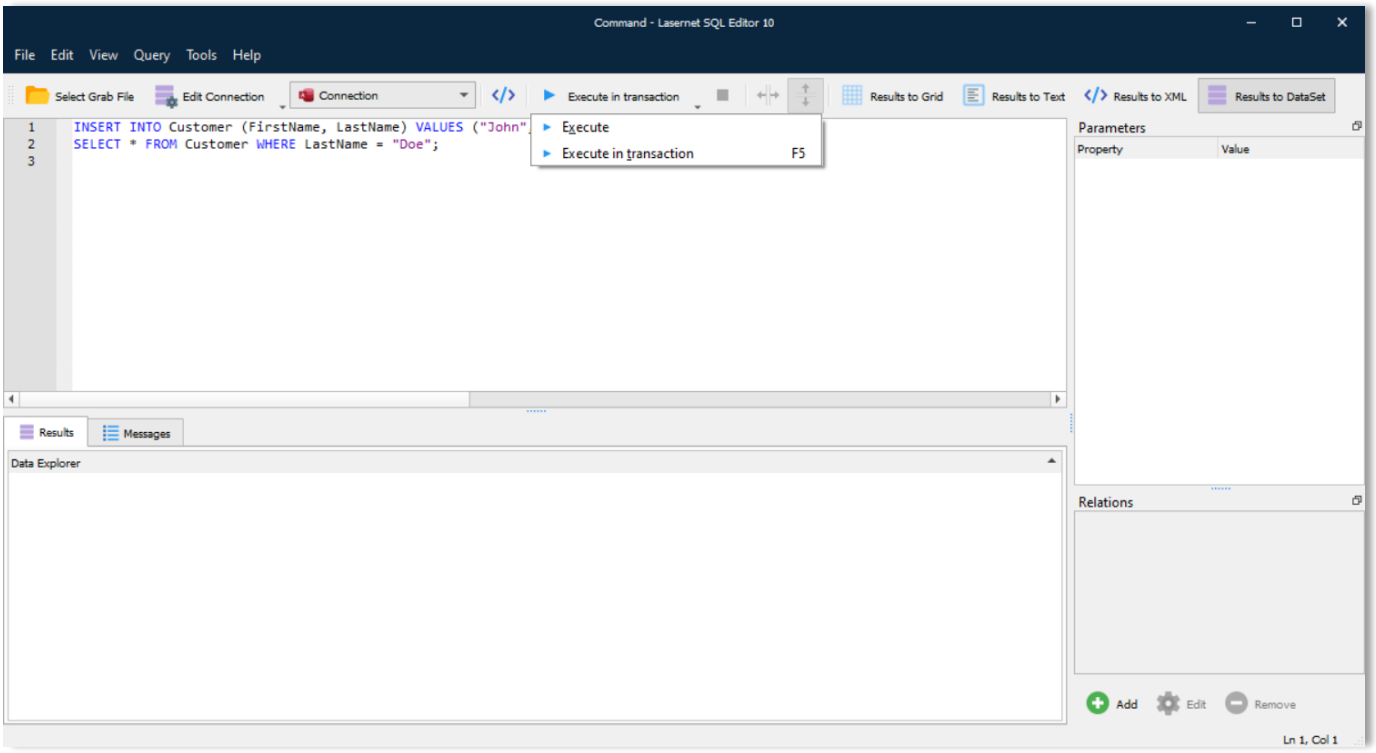




Datasets returned from a stored procedure are either inserted as JobInfos or as XML in JobData exactly like running a ‘Command Text’.

9.4 Executing a query in transaction or without transaction

The **Execute** or **Execute in transaction** button in the tool bar is used for testing queries directly from the SQL Editor.



9.4.1 Execute with transaction

It can be advantageous to execute queries without actually changing the data in the database, especially when testing. As such, this is the default behaviour of the software in order to prevent accidental changes to the database being made. The feature issues a BEGIN TRANSACTION before executing the query and afterwards does a ROLLBACK TRANSACTION. Log and result sets show changes as they were at the time. Multiple changes and selects can be done simultaneously in the same transaction, if each query is terminated with a semi colon.

INSERT INTO Customer (FirstName, LastName) VALUES ("John", "Doe");

SELECT * FROM Customer WHERE LastName = "Doe";

9.4.2 Execute (without transaction)

Execute works identically to Execute with transaction except the BEGIN/ROLLBACK TRANSACTION is not issued. This means changes done to the database are permanent.

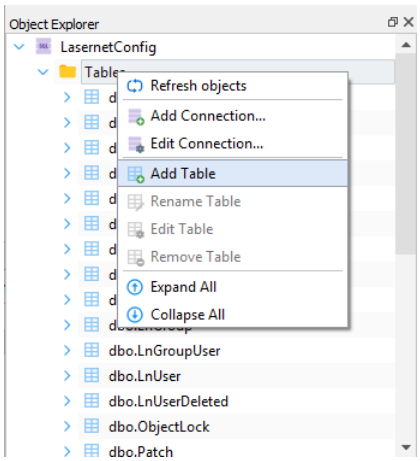
9.4.3 Executing partial queries

When having a several queries separated by semicolon, it can be useful to only execute a few of them. This can be done by selecting just the text covering the queries and then executing the query.

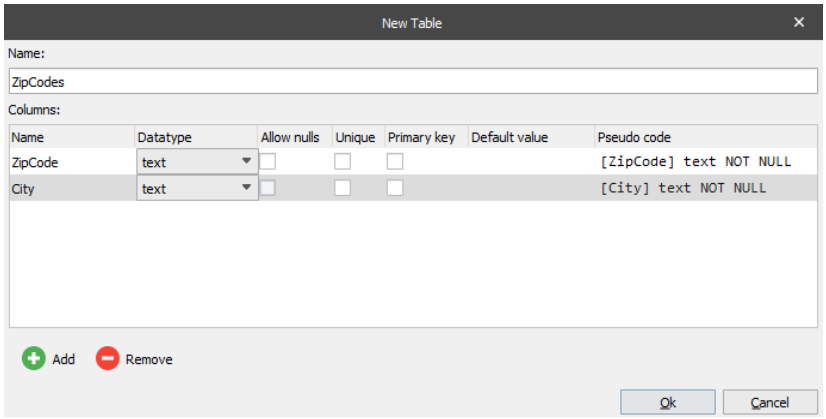
9.5 Table Editor

The table editor is built into the SQL Editor. You need a working connection to a database before you can manipulate tables in it. The table editor is only supported on a small subset of the supported backends in Lasernet.

The Table editor is accessed via the Object Explorer using the Add and/or Edit table option.



The features available in the table editor depend on the backend used. For example, some backends do not support the renaming of columns.



A table name must be specified and at least one column added to the table.

The specified datatype of the columns will reveal or lock out additional features e.g. precision of doubles.

The default value is used when inserting rows in the table where the column is not specified and given a value.

The Pseudo code example is for advanced users who want to know exactly how the column is created (SQL query sent to backend).

Relations between tables cannot be manipulated with the SQL/Table Editor, but must be done either with other software or via manually created SQL queries.

9.6 Database specific issues

9.6.1 MySQL and SQLite

In Lasernet 9, and older, an integration with native database drivers to MySQL and SQLite may require that their DLL's are copied to the Lasernet installation folder.

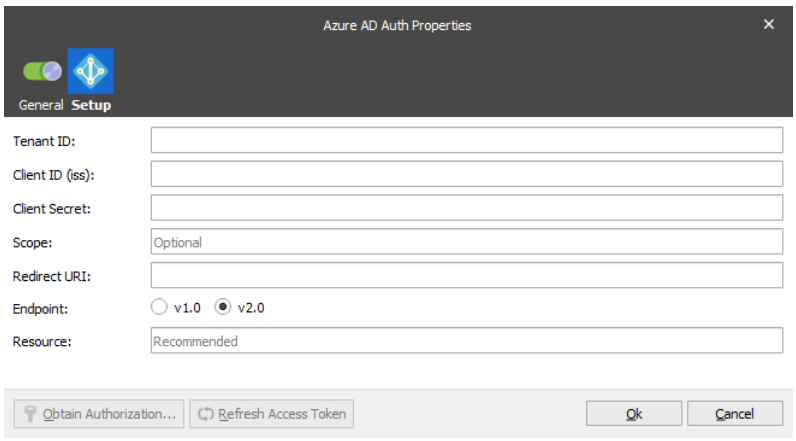
Important: Due to security and synchronizing of Lasernet apps, with Lasernet Server, native database drivers for MySQL and SQLite are no longer supported from Lasernet 10. They can as an alternative be replaced with an Ole DB provider, which must be installed separately.

10 Modifiers.

Several Lasernet modules are available as both modifiers and engines. We recommend using modifiers unless jobs are split (cloned) or merged (combined).

10.1 Azure AD Auth

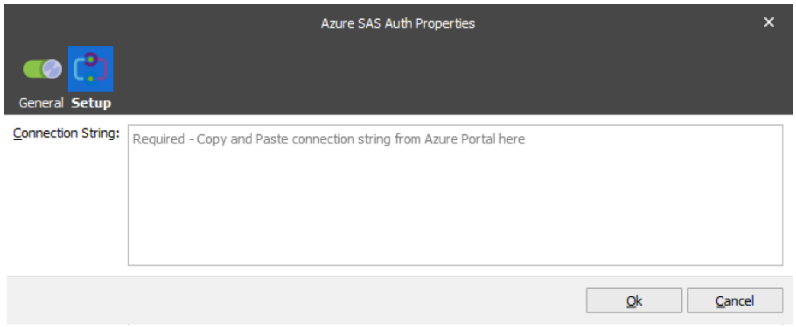
The Azure AD Auth modifier logs into Azure AD for a given user and Azure returns an access token and is used to access different services in Microsoft Azure.



Please refer to the “Lasernet 10 – Azure” manual for more information.

10.2 Azure SAS Auth

Shared Access Signatures (SAS) are the primary security mechanism for Service Bus messaging. The module exchanges a connection string with a token to be used with either the SOAP Web Service module or the HTTP module.



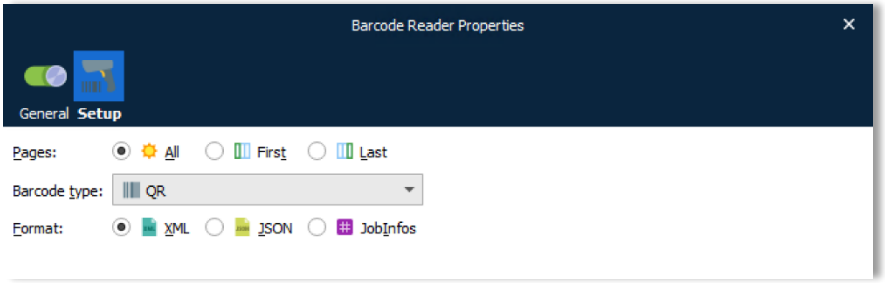
Please refer to the “Lasernet 10 – Azure” manual for more information.

10.3 Barcode Reader

Lasernet can read a wide range of linear and 2D barcode formats in PDF documents, PNG, JPEG and TIFF images.

Some common barcode types that can be read by this module are:

- Code 128
- Code 39
- Data Matrix
- EAN-13
- QR



10.3.1 Pages

Select the pages of the document you want to be scanned for barcodes.

All – All pages in the document will be scanned for barcodes.

First – Only the first page the document will be scanned for barcodes.

Last – Only the last page the document will be scanned for barcodes.

10.3.2 Barcode type

Select, from the drop-down menu, the barcode type for which you want to scan:

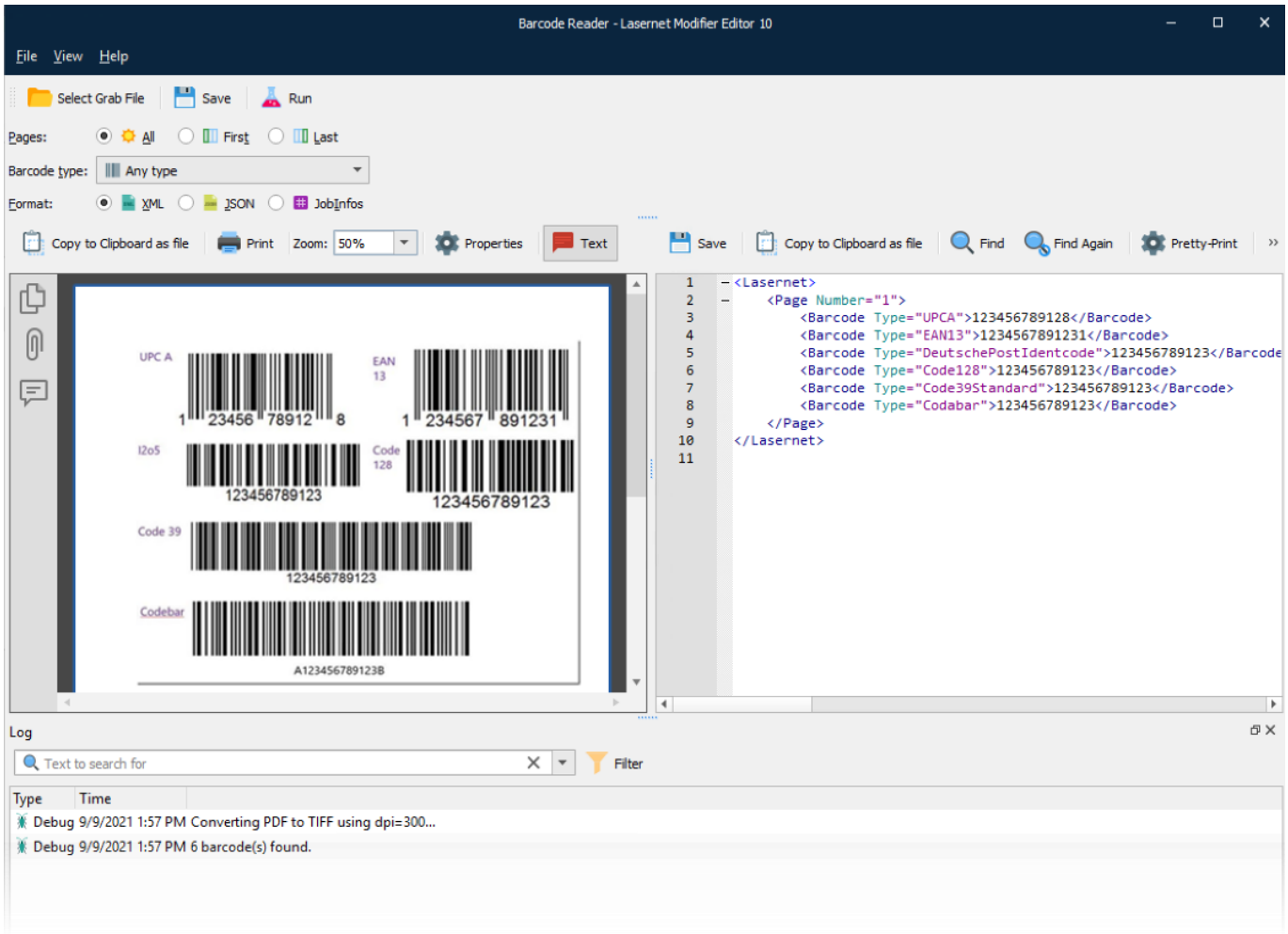
- One particular barcode type
- All supported barcode types

Note: We recommend scanning a document for one particular barcode type if possible because processing time will be reduced compared to scanning for all barcode types.

10.3.3 Format

The extracted output will be formatted according to the selection:

XML – The barcode read data will be presented in XML format.

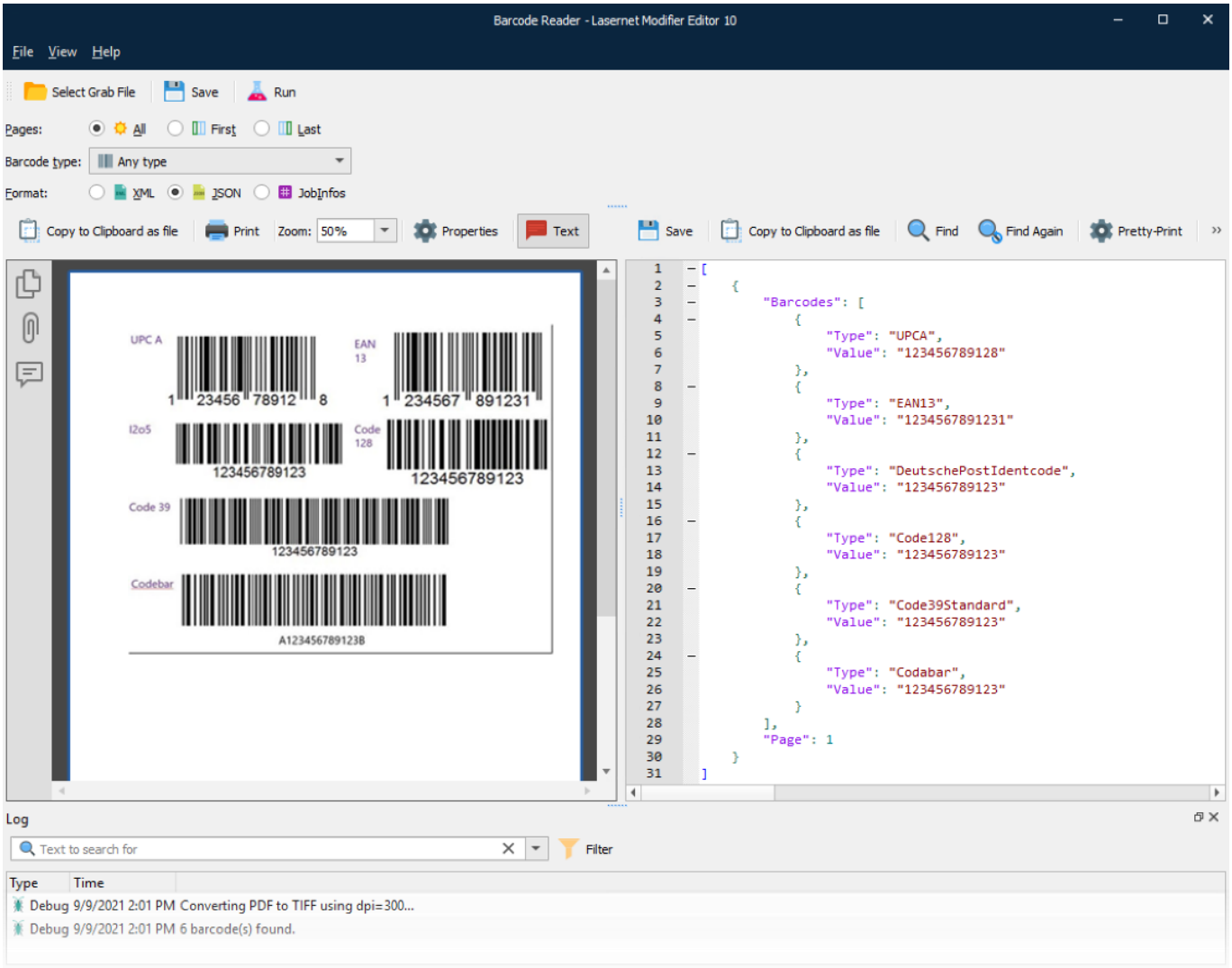


No additional JobInfos are created by the modifier.

The following information is captured and presented in XML format:

- Page
- Barcode Type
- Barcode Value

JSON – The barcode read data will be presented in JSON format.

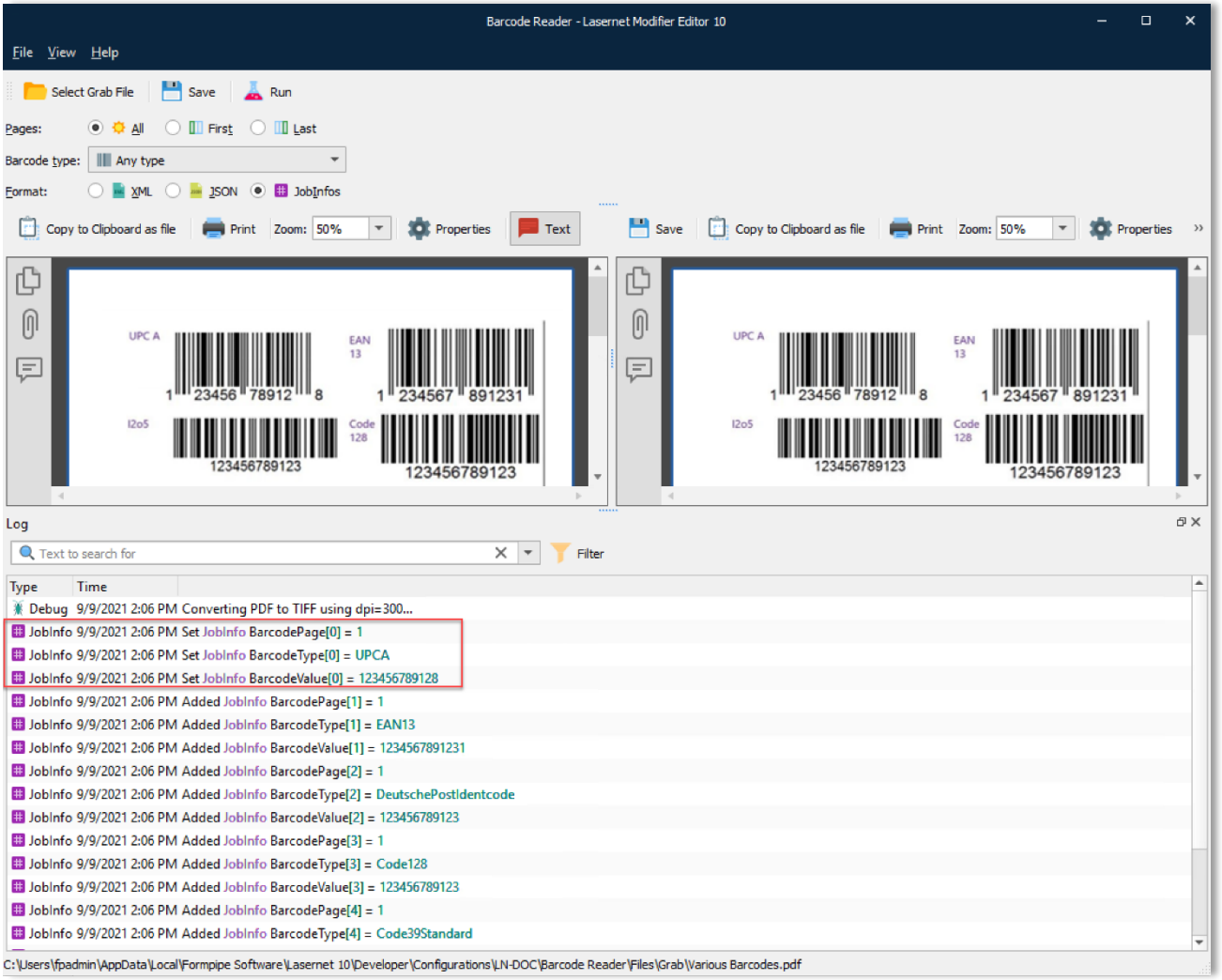


No additional JobInfos are created by the modifier.

The following information is captured and presented in JSON format:

- Page
- Barcode Type
- Barcode Value

JobInfos – The barcode read data will be presented in the Lasernet native format.



PDF output has an array of JobInfos.

There are names and values for:

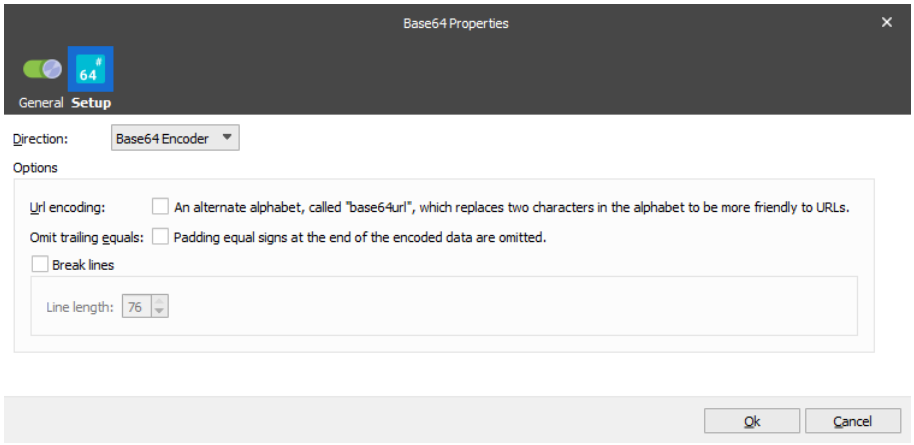
- BarcodePage
- BarcodeType
- BarcodeValue

10.4 Base64

The Base64 modifier is used to encode/decode base 32 or base 64 data. This encoding method is used for converting binary (8-bit) data to 6-bit ASCII-printable data.

10.4.1 Settings

Direction	Base32 Encode/Decode, Base64 Encode/Decode.
URL encoding	Use an alternate alphabet, called "base64url", which replaces two characters in the alphabet to be more URL friendly. The default is the regular Base64 alphabet, called simply "base64".
Omit trailingequals	Omits adding the equal sign padding at the end of the encoded data. The default is to keep the trailing equal sign padding at the end of the encoded data, so the data is always a size multiple of four.
Break lines	Split the string by a specified number of characters. Useful with MIME which enforces a limit on line length of base 64-encoded data to 76 characters.



10.5 Binary Filter

The modifier works by searching for a regular expression. For each match, it will replace it with another string.

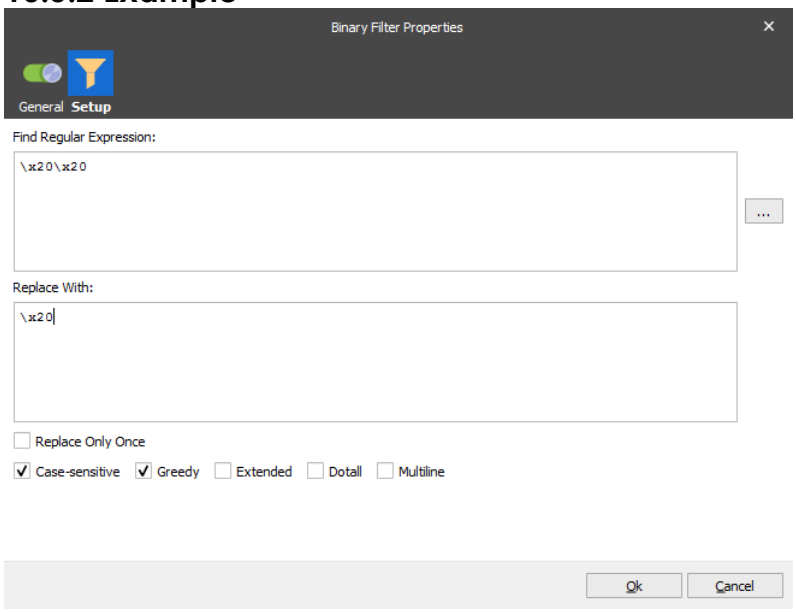
Both the regular expression and the replace text are written in the Latin-1 (ISO-8859-1) character set. However, because this modifier is intended for binary searches, most (if not all) characters should be written in hex using the \x escape.

The 'replace' text field supports the $1 $2… $n markers, to include portions of matched regular expressions in the replacement. This provides the ability to create fairly advanced filters, e.g., if the regular expression is '(\d+)\x20(\d+)' and the replacement string is '$2\x20$1', then the filter will search for two numbers separated by a space character and swap the numbers.

10.5.1 Settings

Find Regular Expression	Regular expression to search for.
Replace With	The search value to be replaced, typed as a string or hex value. For example, \x0d\x0a.
Replace Only Once	If the check box is set, it will only replace the first match. If it is not checked, it will replace all matches.

10.5.2 Example



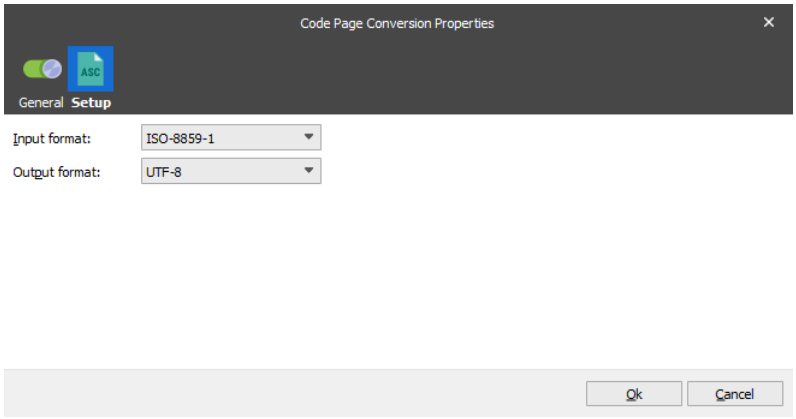
This example searches for two spaces and replaces it with one space.

10.6 Code Page Conversion

Used for converting from one code page to another.

10.6.1 Settings

Only two settings are applicable for this modifier: the Input format and the Output format. Both are chosen using drop-down lists.

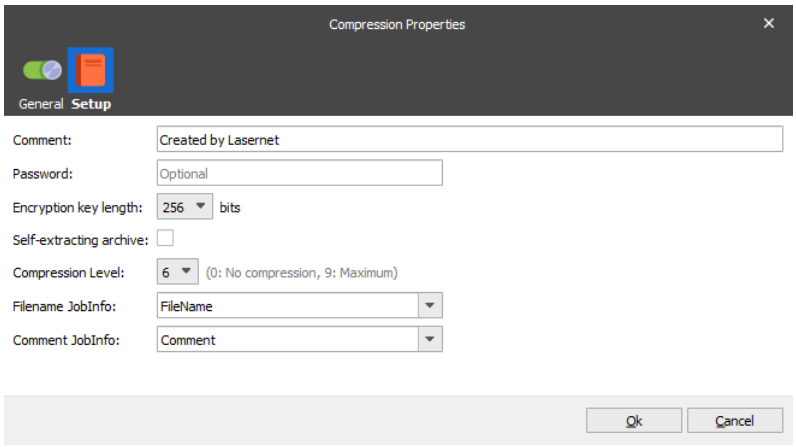


The modifier sets the JobInfo *ActiveCodePage* to the value of *Output format*.

Please note that even though you are able to choose some of the code pages when editing the modifier in Lasernet Developer, these code pages may not necessarily exist on the server.

10.7 Compression

The modifier is able to compress the content of JobData into a ZIP archive or self-extracting archive (no other compression archives are supported).



If more than one Job needs to be compressed into a single ZIP archive, combining must be enabled.

10.7.1 Settings

A range of properties can be set to control the resulting ZIP archive.

Comment	Comment to be embedded in ZIP archive
Password	If the ZIP archive needs to be encrypted, a password is required.
Encryption key length	Length of the encryption key.
Self-extracting archive	Lasernet generates a self-extracting executable rather than a ZIP archive. Extension of the file will be set to ".exe".
Compression level	The higher the value, the smaller is the ZIP file. A higher value takes a longer time. Lower values should not be used unless a real performance problem exists. Can be overruled at runtime via JobInfo CompressionLevel.
Filename JobInfo	JobInfo containing filename to be included in ZIP archive

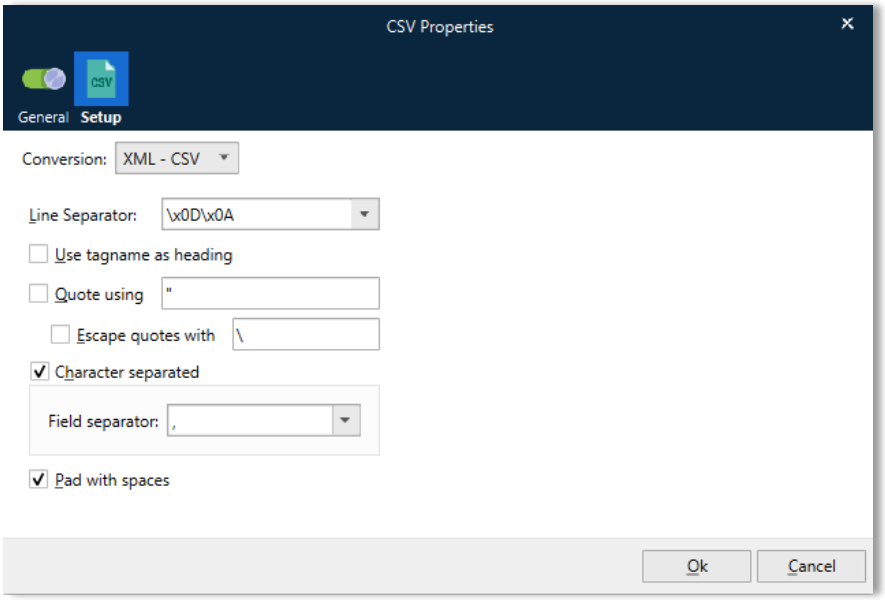
JobInfo FullFilename must be specified. Paths are supported, excluding the drive letter.

10.8 CSV

The CSV modifier is used for transforming XML (eXtended Markup Language) to a CSV file (comma separated) and vice versa. The modifier is very useful for allowing modern systems to interface with legacy systems.

The modifier is capable of producing both column and character separated files, plus XML (Lasernet defined format) from CSV.

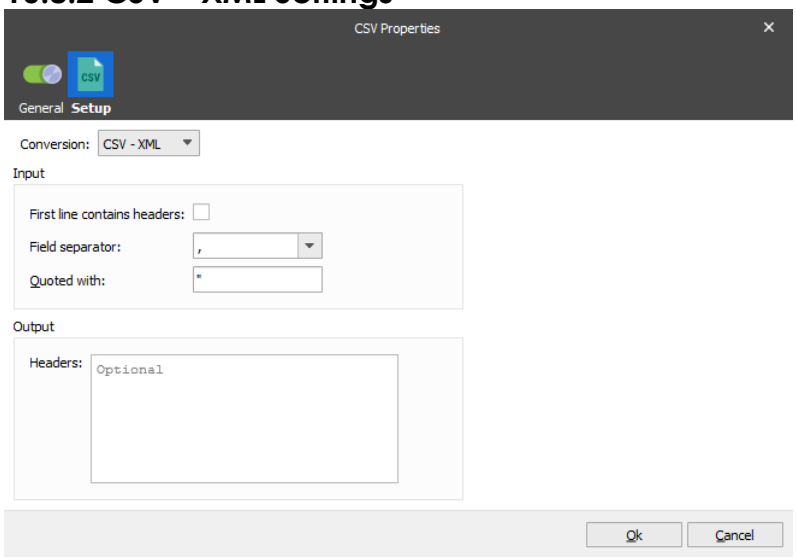
10.8.1 XML – CSV Settings



Conversion	Choose between converting between XML to CSV and CSV to XML.
Line Separator	Determines what line separator to use between lines in the text file.
Use tagname as heading	If checked, the modifier will create a header line where the names of the tags are used for naming the columns (see below).
Quote using	If checked, a quote character should be specified, usually “, that the modifier will put around the text of each column. Note, that when using quotes the column width must be adjusted accordingly since the quotes are not counted in the width of the column.
Escape quotes with	If checked, the modifier will search the text in each column for the character entered in ‘*Quote using’* and escape it with the character specified.

Character separated	If checked, the modifier produces character-separated files. Enter the column separator e.g. comma, semicolon colon or tab.
Pad with spaces	When checked, the modifier pads the text in the columns with spaces until they fill out the width determined by the xml-file.

10.8.2 CSV – XML Settings



Conversion	Choose between converting between XML to CSV and CSV to XML.
First line contains headers	Often the first line in the CSV contains the headers of the fields. Checking this option will make Lasernet use the header names in the XML, plus skip the first line as it does not contain data.
Field separator	Defines the separator between the fields in the CSV.
Quoted with	Tells Lasernet that fields in the CSV are quoted and with which character. This is useful when a field separator occurs within a field.
Headers	Optionally set the header names if not included in the CSV or if they are not correct. If not in the CSV or specified here, the fields will be named by Lasernet.

10.8.3 XML format

The modifier expects its XML-input to be in a certain format. The same format is used for XML-output.

```
<table>
  <ItemLine visible=”0”>
    <Field1 width=”20” alignment=”left”>field value</Field1>
    <Field2 .....
  </ItemLine>
  <ItemLine>
     ....
  </ItemLine>
</table>
```

The name of the root tag (table) does not matter. For each line in the resulting text file, there must be one set of *ItemLine* tags.

The visible attribute can be used to determine whether the item should be put into the text file or not. If it is missing, the default value is *1*.

For each *ItemLine,* there are a number of fields which contain the column data. For column separated files, the *width* attribute is used for setting the width of the column. If this attribute is missing, the modifier assumes the default value of *20*.

The *alignment* attribute can be set to either *left* or *right*. If missing, *left* is assumed.

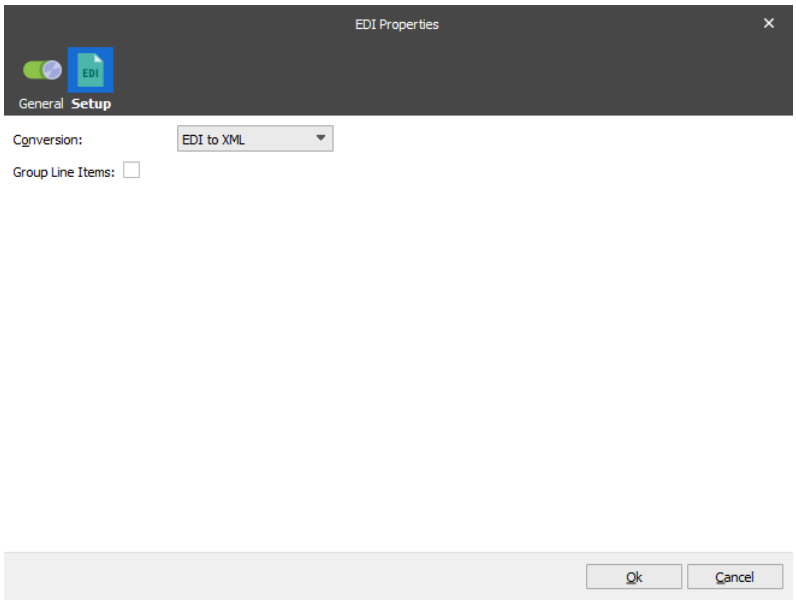
Use tag names of the files for the header line, if desired.

10.9 EDI

This modifier was previously used to convert an EDI file to XML or an XML template to EDI. However, as from Lasernet 8, EDI support was implemented as an ingoing and outgoing business format directly from the Lasernet Form Engine/Editor.

10.9.1 EDI to XML

Define the default EDI (UNA) characters for escaping, grouping, separating and segmenting that will be included in the EDI file, for successful conversion into XML. UNA characters are typically located in the first line of EDI data. Setting decimal characters currently has no effect and is included for future releases.



Example of EDI input:

```
UNA:+.? '
UNB+
UNOC:3+007007007:14+FLOORBALL:ZZZ+050601:0339+104'
UNH+1+INVOIC:D:96A:
UN:EAN008'
BGM+380+1234567890'
DTM+137:20050601:102'
FTX+AAB+++text 1'
FTX+AAB+++text 2'
FTX+AAB+++text 3'
FTX+AAB+++text 4'
RFF+VN:33333'
RFF+DQ:44444'
DTM+171:20050515:102'
NAD+II+77777777::91+name and addr line+party name+street 1+city++12345'
UNS+S'
MOA+176:21.82'
MOA+86:158.19'
MOA+125:136.37'
UNT+55+1'
UNZ+1+104'
```

Example of XML Output:

```
<EDI>
 <UNA/>
 <UNB>
  <group>
```

```
  <value>UNOC</value>
  <value>3</value>
 </group>
 <group>
  <value>007007007</value>
  <value>14</value>
 </group>
 <group>
  <value>FLOORBALL</value>
  <value>ZZZ</value>
 </group>
 ...
</UNB>
<UNH>
 <group>
  <value>1</value>
 </group>
 <group>
  <value>INVOIC</value>
  <value>D</value>
  ...
 </group>
</UNH>
<BGM>
 <group>
  <value>380</value>
 </group>
 <group>
  <value>1234567890</value>
 </group>
</BGM>
<DTM>
 <group>
  <value>137</value>
  <value>20050601</value>
  <value>102</value>
 </group>
</DTM>
<FTX>
 <group>
  <value>AAB</value>
 </group>
 <group>
  <value></value>
 </group>
 <group>
  <value></value>
 </group>
 <group>
  <value>text 1</value>
 </group>
</FTX>
...
```

```
<RFF>
 <group>
  <value>VN</value>
  <value>33333</value>
 </group>
</RFF>
...
<DTM>
 <group>
  <value>171</value>
  <value>20050515</value>
  <value>102</value>
 </group>
</DTM>
<NAD>
 <group>
  <value>II</value>
 </group>
 <group>
  <value>77777777</value>
  <value></value>
  <value>91</value>
 </group>
 ...
</NAD>
<UNS>
 <group>
  <value>S</value>
 </group>
</UNS>
<MOA>
 <group>
  <value>176</value>
  <value>21.82</value>
 </group>
</MOA>
...
<UNT>
 <group>
  <value>55</value>
 </group>
 <group>
  <value>1</value>
 </group>
</UNT>
<UNZ>
 <group>
  <value>1</value>
 </group>
 <group>
  <value>104</value>
 </group>
```

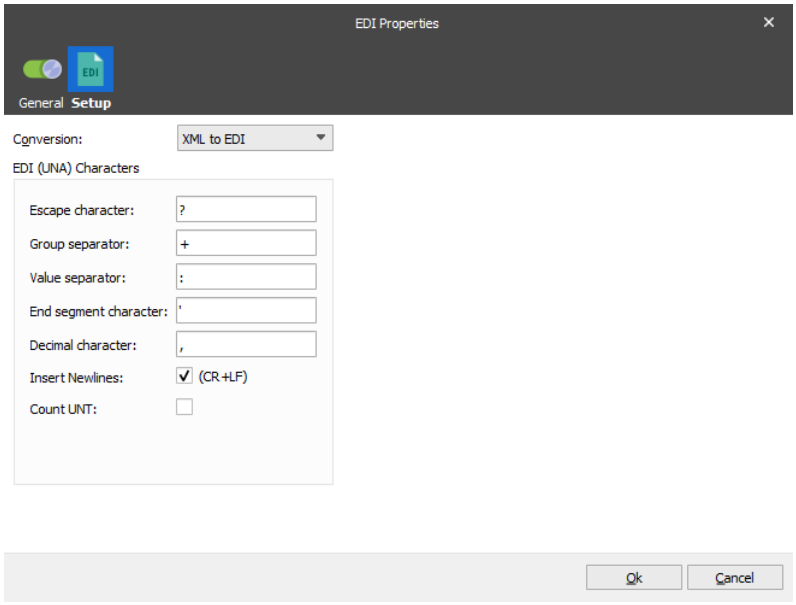
</UNZ>
</EDI>

10.9.2 XML to EDI

Define the **EDI (UNA) characters** for escaping, grouping, separating and segmenting that will be embedded into the EDI file after conversion has taken place. Insert Newlines can be activated for inserting newlines after each line in the XML file.

Activate **Insert Newline** to add a newline after each segment, or deactivate to create an EDI file without newlines.

Activate **Count UNT** to calculate and include the number of segments in the EDI file. The UNT segment must already be present in the message trailer to function correctly.



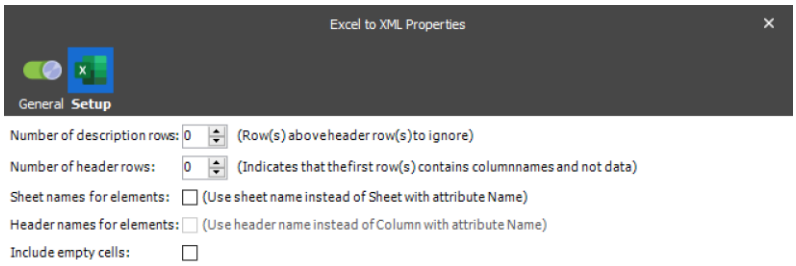
For successfully converting XML to EDI, the XML must include valid grouping, separations and segments. Only XML files created via the EDI to XML modifier will have a valid structure by default. In the Form Editor the XML file is very useful as a template for creating a valid EDI file based on the EDI to XML modifier (XML to EDI mode).

More information about how to work with EDI as input or output is available from the Lasernet Form Editor Guide in the Formpipe Knowledge Base. The EDI format is now built in and replaces the need for the EDI modifier (previously used in older versions of Lasernet).

10.10 Excel to XML

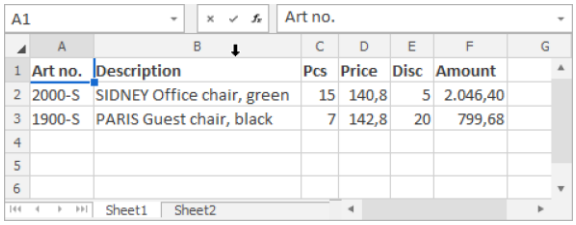
The Excel to XML modifier is used for converting a Microsoft Excel spreadsheet into an XML structure. In the list of modifiers, “Excel to XML” must be added and the properties for the conversion must be set.

Spreadsheets created by Microsoft Excel 2007 or newer are supported without any dependencies to Microsoft Excel.



10.10.1 Data fields

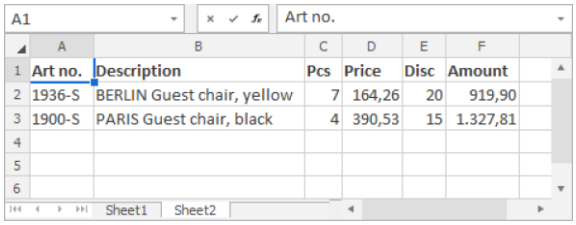
Rows and columns on all sheets are extracted and saved into the XML structure.



	A	B	C	D	E	F	G
1	Art no.	Description	Pcs	Price	Disc	Amount	
2	2000-S	SIDNEY Office chair, green	15	140,8	5	2.046,40	
3	1900-S	PARIS Guest chair, black	7	142,8	20	799,68	
4							
5							
6							

Sheet1 Sheet2

Above is an example of the first sheet named Sheet1, containing a header and two rows.



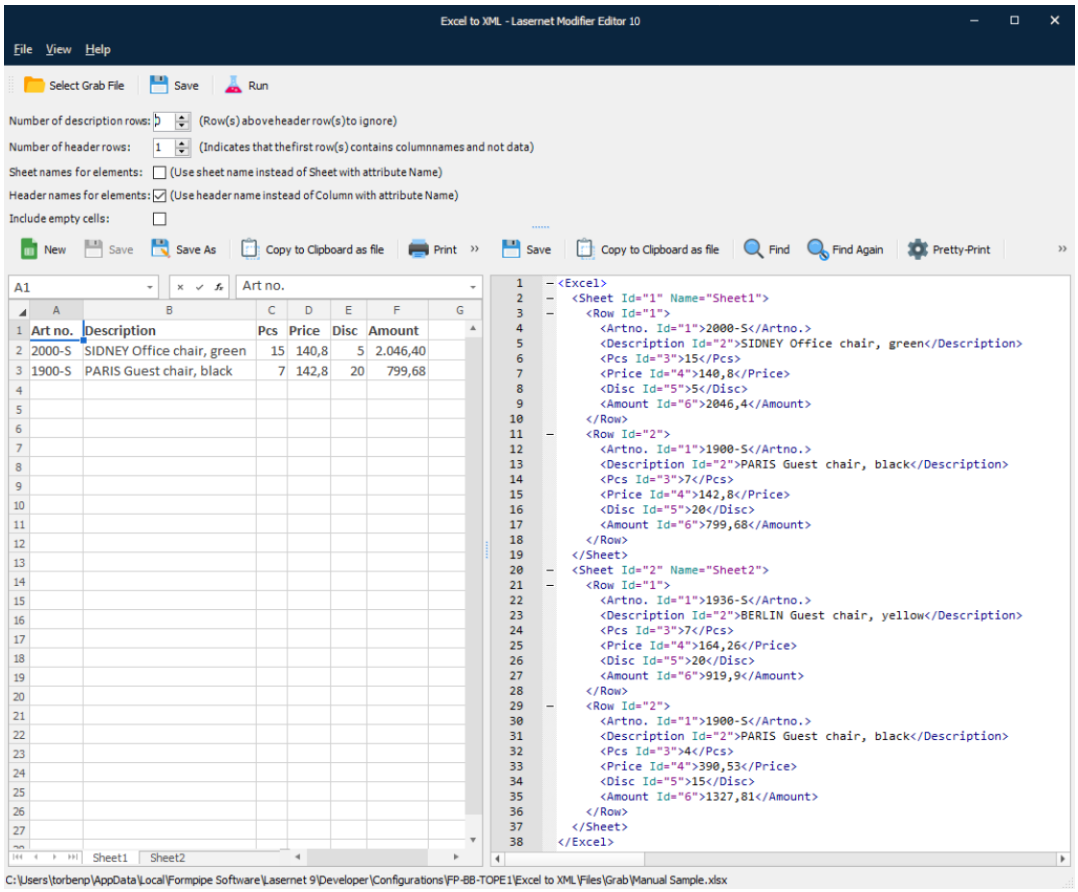
	A	B	C	D	E	F
1	Art no.	Description	Pcs	Price	Disc	Amount
2	1936-S	BERLIN Guest chair, yellow	7	164,26	20	919,90
3	1900-S	PARIS Guest chair, black	4	390,53	15	1.327,81
4						
5						
6						

Sheet1 Sheet2

Above is an example of a second sheet named Sheet2, containing a header and two rows.

The final result is:

- An XML file containing a root node <Excel>
- A child for each sheet named <Sheet> with attributes for @Id = Sheet index, @Name = Sheet name
- Sub node for each row named <Row> with attribute for @Id = Row index
- Sub node for each column named <Column> with attributes for @Id = Column index, Name = Header Row (if available/selected)



Above is the example XML file containing the two sheets with header, rows and columns.

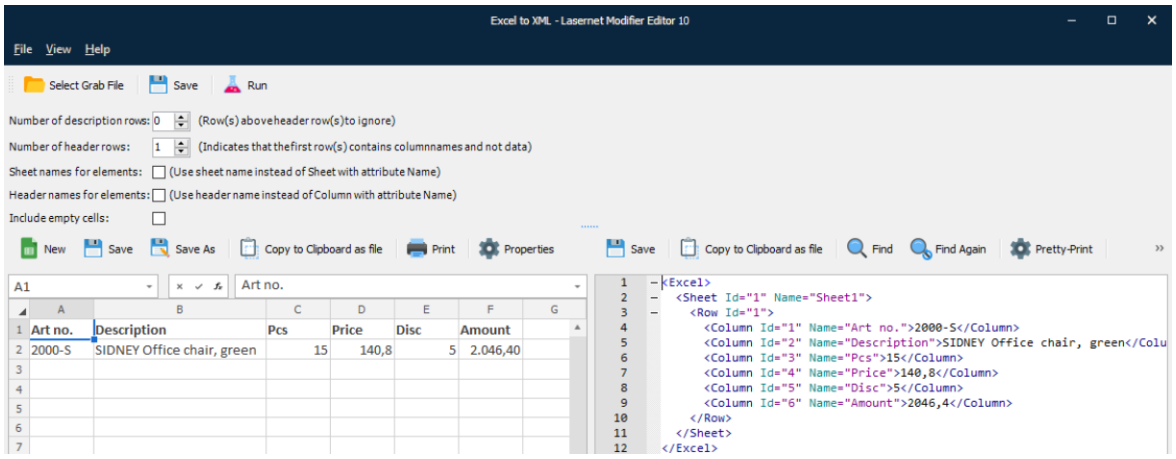
10.10.2 Number of description and Number of header rows

The **Number of description rows** defines a number of rows to ignore before the first row is extracted from the spreadsheet.

Number of header rows setting will affect the final structure of the generated XML file. If set to 0 (zero), it indicates that the spreadsheet has no header and represents the first row in data.

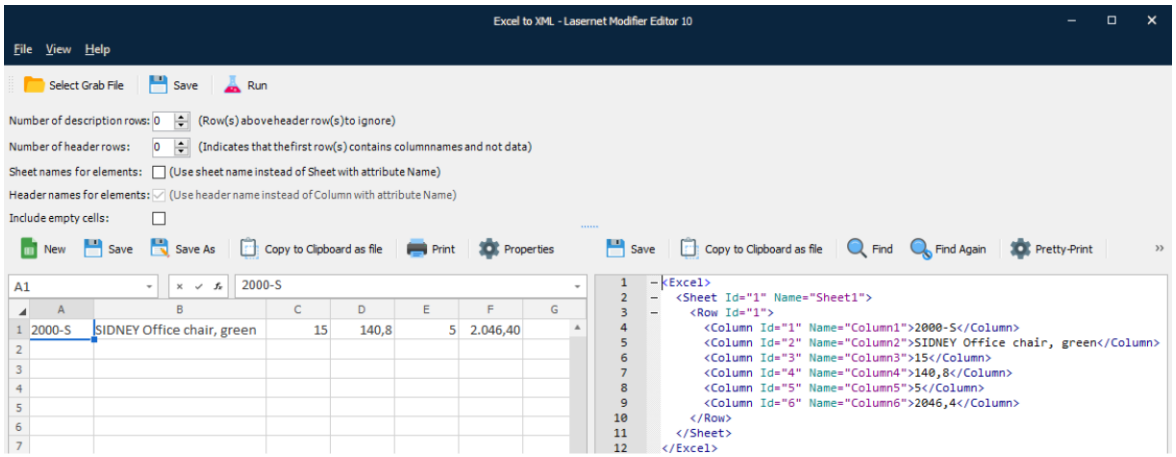
It is used for determining if the first row in the Excel spreadsheet should be parsed as column names and not as data.

Value is 1 or higher: Indicates that the first row contains columns and not data



Excel sample with header information and XML sample with header information included as an attribute.

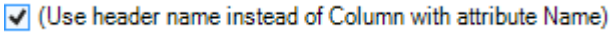
Value is 0 (zero): Indicates that the first row contains data and not columns



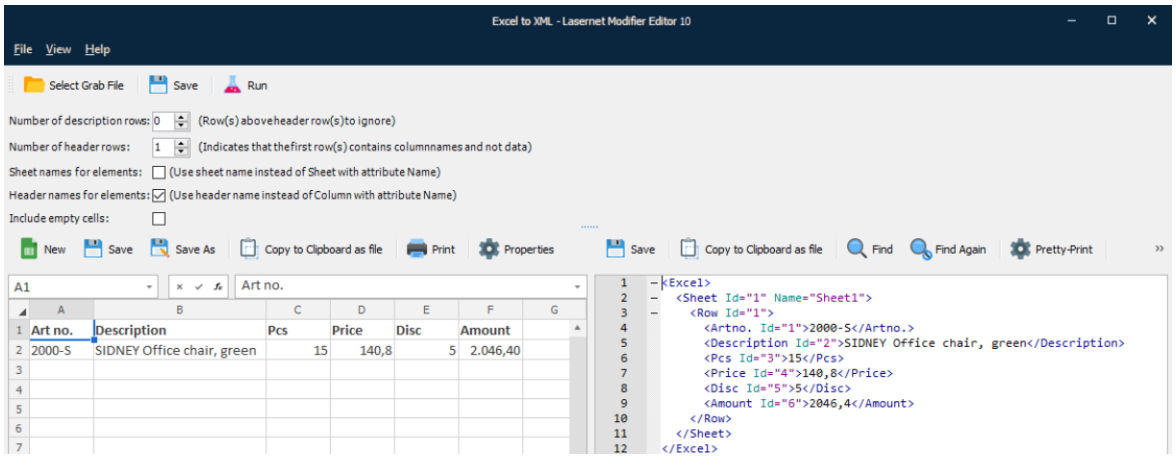
Excel sample without header information and XML sample without header information.

10.10.3 Sheet name and Header names for elements

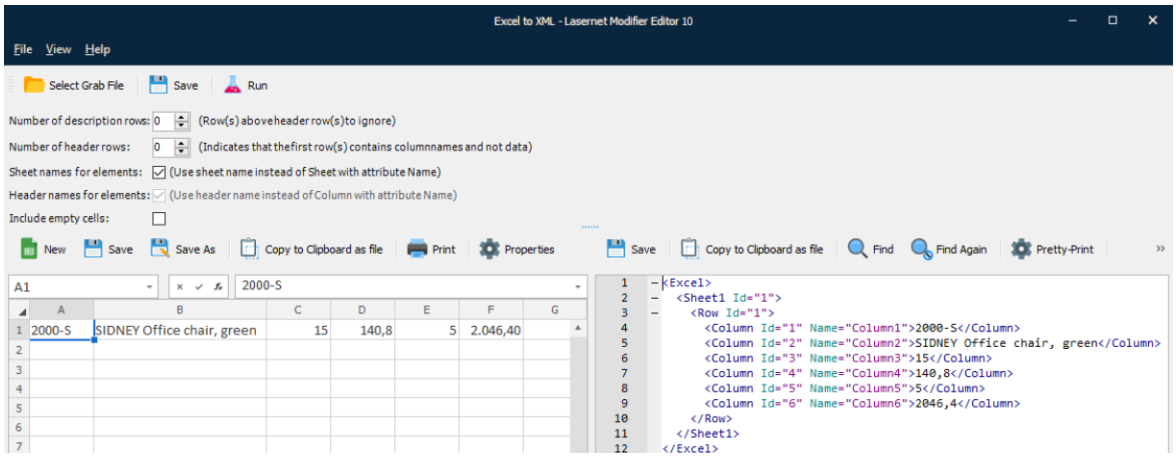
Sheet names for elements and **Header names for elements** settings will affect the final structure of the generated XML file. The **Header names for elements** is only active if number of headers rows is larger than 0 (zero).



If activated the header name for the first row is used as element name.



Excel sample with header information and XML sample with header information included as element names.



Excel sample with header information and XML sample with sheet and header information included as element names.

10.10.4 Include empty cells

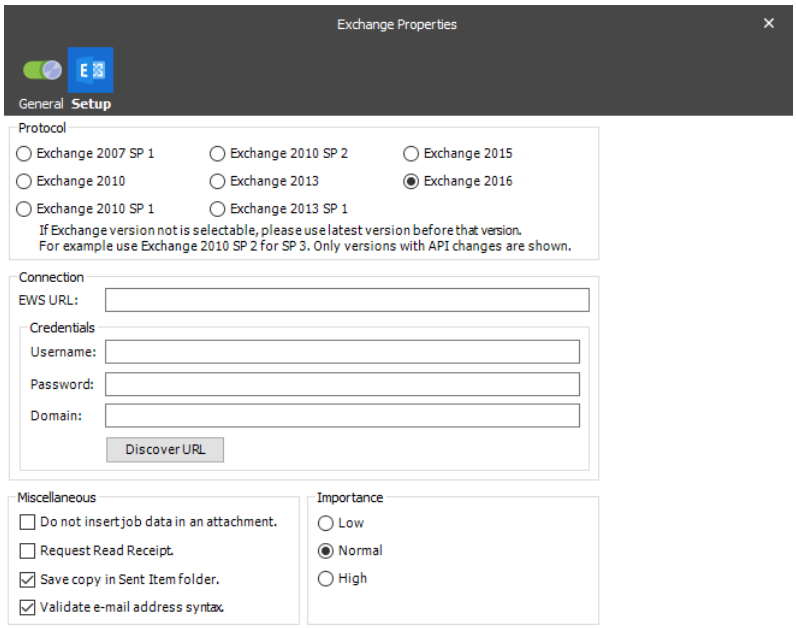
If the **Include empty cells** setting is activated the XML data will also include the cells with empty values otherwise, they will be ignored and not present in XML.

10.10.5 Unsupported features

The Excel to XML modifier does not support protected spreadsheets and the output result will be empty.

10.11 Exchange

Used for sending emails via Microsoft Exchange Server.

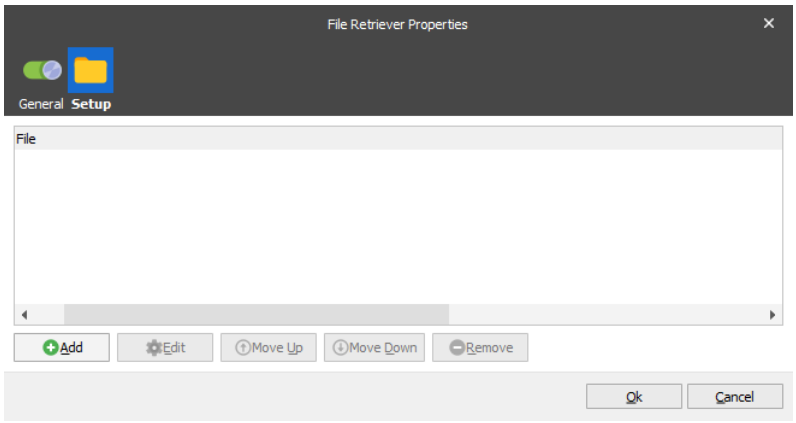


10.11.1 Settings

See section for Exchange output module for more information.

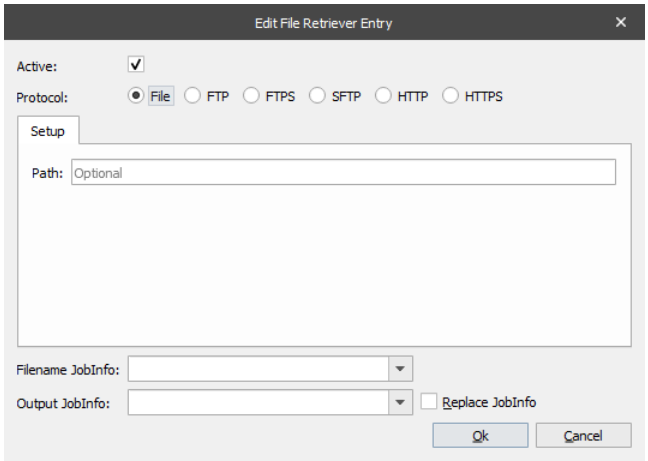
10.12 File Retriever

Whilst processing a job, this modifier can retrieve additional files from different locations like a file directory, FTP-server or HTTP-server, in either normal or secure mode.



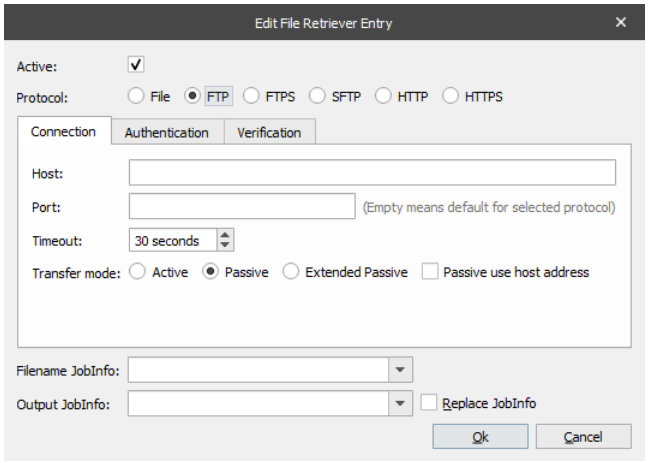
10.12.1 Settings

You can add a list of locations to retrieve your additional files from, based on a filename stored in a user defined JobInfo. If the file is successfully retrieved from the specified location the data will be placed into a result output JobInfo.

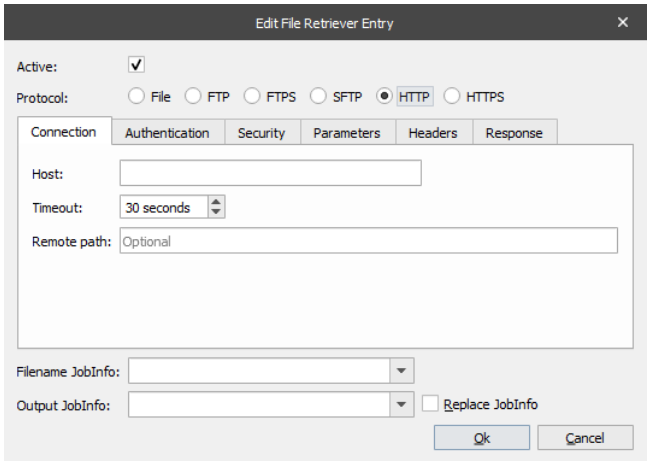


If the file you want to retrieve is stored on a file system, you must configure the following settings:

Protocol	Data will be retrieved via the specified protocol (File, FTP, HTTP) in secure or non-secure mode.
Path	The location where your files are stored.
Filename JobInfo	The name of the JobInfo containing your file name.
Output JobInfo	The name of the JobInfo where you want to store your data.
Replace JobInfo	If active, any retrieved files will be added to index 0.



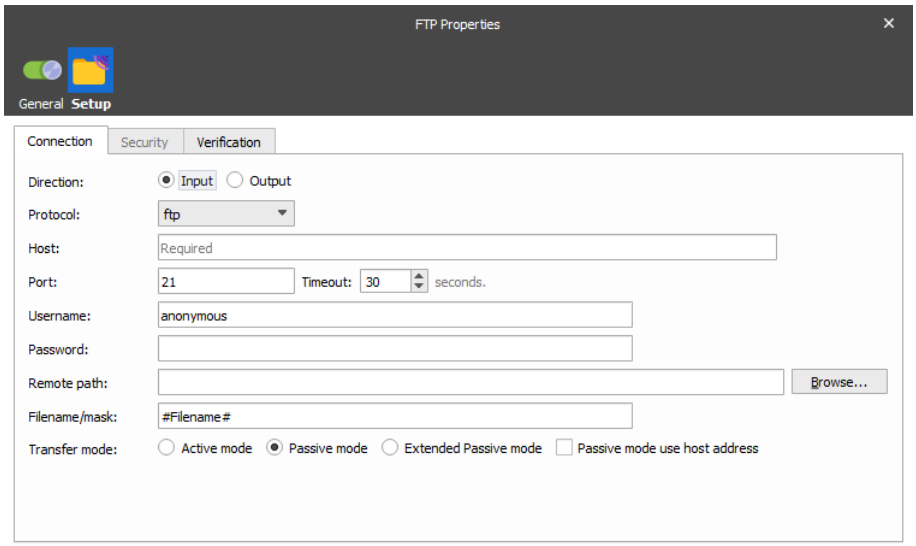
If the file you want to retrieve is stored in an FTP or HTTP location, you must specify additional settings for host name, port number, path etc. For more detailed information about FTP and HTTP protocols search the internet.



NOTE: The File Retriever is developed as a replacement for the File Database engine, with support for several protocols and an improved / easier way to define references for files to be retrieved. Therefore, we recommend using the File Retriever where possible.

10.13 FTP

Whilst processing a job, this modifier is used to send or retrieve files from an Internet URL destination such as an FTP, FTPS or SFTP server.



10.13.1 Settings

See section for FTP Input and Output modules for more information.

10.14 HTML to XHTML

Converts HTML to the more structured and precise XHTML format. It can fix many common errors in HTML files (e.g. missing end tags, elements with incorrect content model, non-standard elements or attributes, etc.) It can also fix invalid or badly-formed HTML, and clean it to produce a well-formed and valid XHTML.

Examples of fixes it can make to bad HTML:

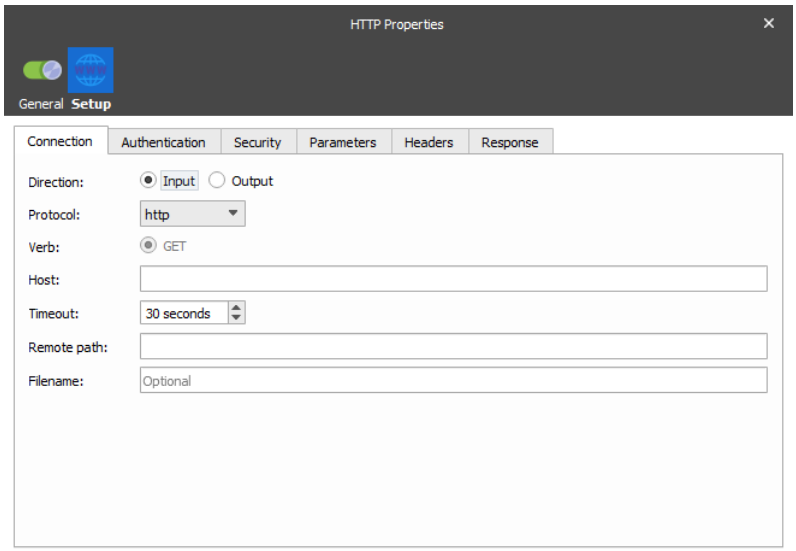
- Straighten mixed-up tags
- Fix missing or mismatched end tags
- Add missing items (some tags, quotes, ...)

10.14.1 Settings

There are no user settings available for this modifier.

10.15 HTTP

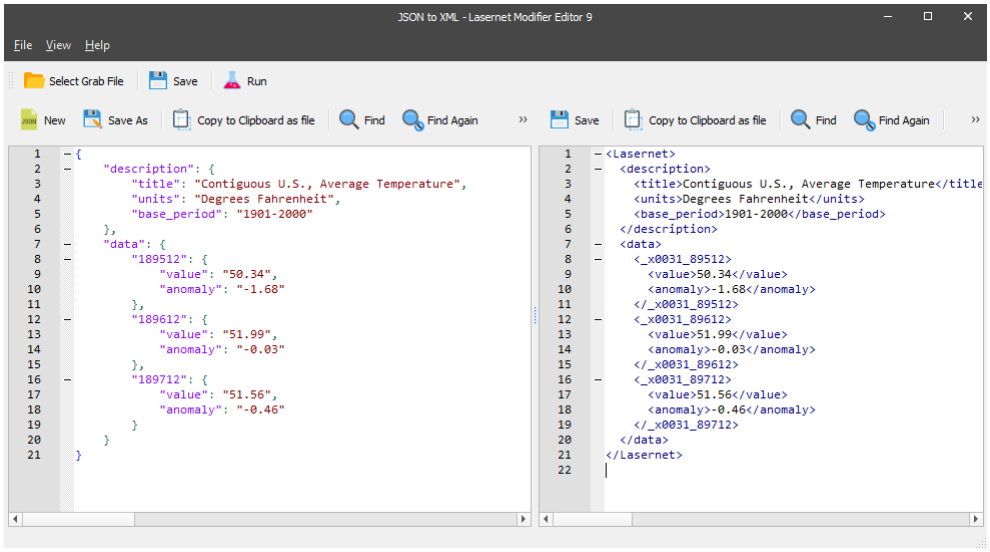
Whilst processing a job, this modifier is used to send to or retrieve files from HTTP and HTTPS servers.



10.15.1 Settings

See section for HTTP Input and Output modules for more information.

10.16 JSON to XML



No settings are available for this modifier. If the JSON format is not validated as a valid format, it will not be converted and output result is equal the input result.

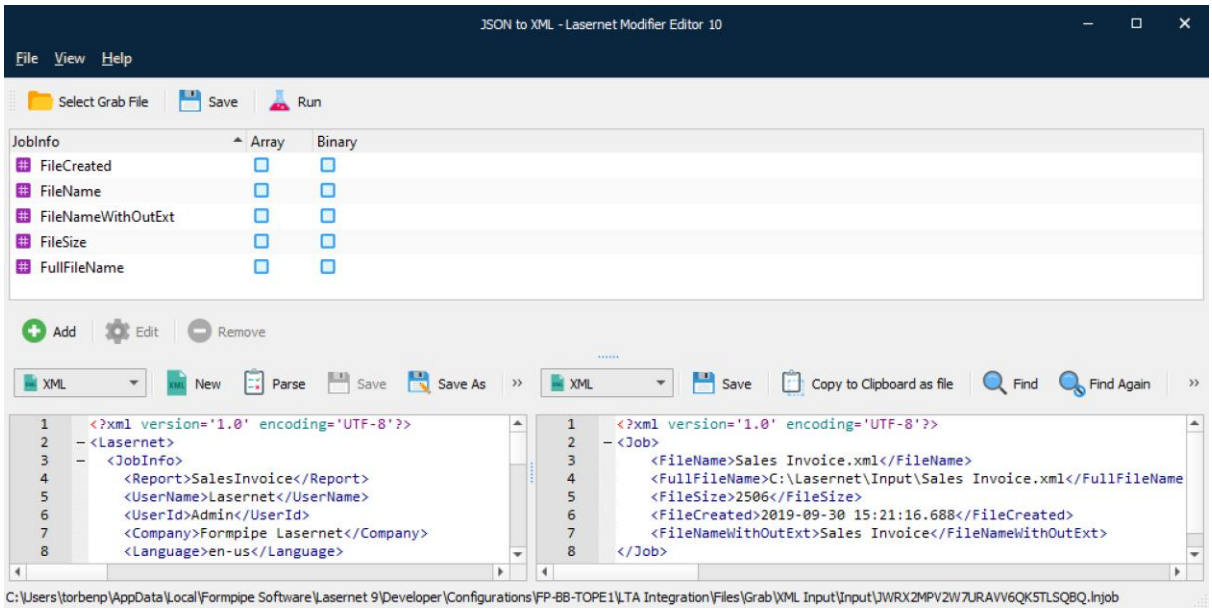
Both JSON and XML can be used to receive data from the web server, but with the JSON to XML converter you can convert the JSON format into an XML structure that can make advantage of the XML Transformer or other XML modules available in Lasernet.

The XML output result is prefixed to start with an <Lasernet> element name and does not include line separators.

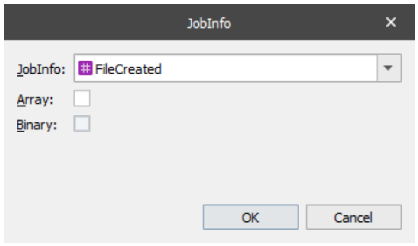
If a name in the JSON format contains a non-valid name for an XML structure it will be prefixed with _x0031.

10.17 Job to XML

Converts a list of selected JobInfos to an XML structure. The values of the JobInfos are retrieved from the job being processed.



Click **Add** to insert a JobInfo you want to include in the XML result. Click **Edit** or **Remove** to maintain JobInfos and values already added to the list.



JobInfo — Name of **JobInfo** and **Value** to be included in the XML result.

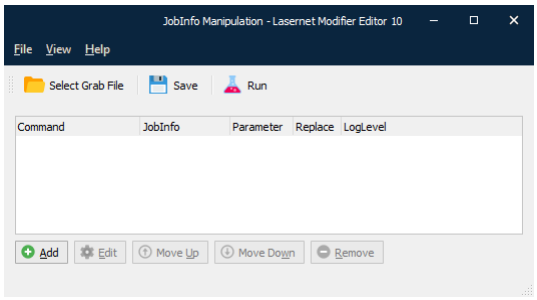
Array — Activate if the JobInfo contains an array of values and all values are required in the XML result. The values in the array of a JobInfo are added as children in an element named **Value**:

```
<JobInfoName>
  <Value>1</Value>
  <Value>2</Value>
</JobInfoName>
```

Binary — Activate if the value of the JobInfo represents a binary value. The value is encoded and included as a base64 string.

10.18 JobInfo Manipulation

This modifier is used to execute any number of commands on one or more JobInfos.



10.18.1 Settings

Determine on Modifier Point / This JobInfo

If Determine on Modifier Point is selected, the Modifier uses the JobInfo that is selected when running the actual instance of the Modifier. If This JobInfo is selected, the particular JobInfo must be entered/selected in the combo box.

Replace existing

If checked, all existing values of the JobInfo chosen will be deleted before adding the new JobInfo value. If not checked, it will be appended to the array of values instead. The replace option does not necessarily make sense for all types of manipulation.

Value

The value parameter has different uses for each manipulation. Generally, it uses standard JobInfo Substitution for calculating the value. See the manipulation list below for details.

10.18.2 Manipulation Commands

append

Appends (places at the end) the value of the parameter. The parameter value is passed through standard JobInfo Substitution before being appended.

assign

Assigns the given parameter value to the chosen JobInfo. The parameter value is passed through standard JobInfo Substitution before being assigned.

binary to hex

Interprets the value of the parameter as binary data and converts it to hexadecimal digits.

combine

Combines the individual array values of the JobInfo into one string divided by whatever the parameter evaluates to, e.g. a semicolon.

delete

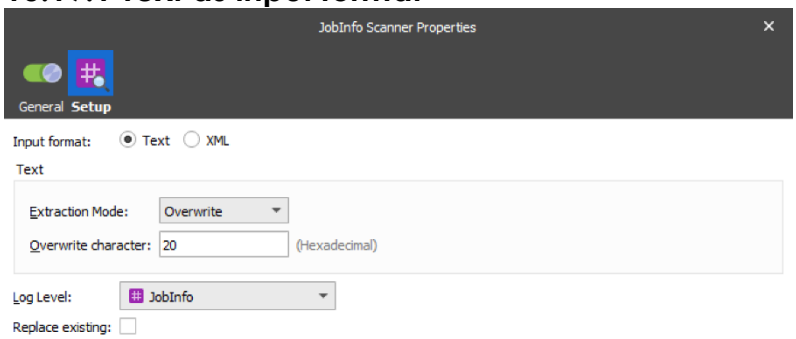
Deletes the given JobInfo. If no parameter is given, the entire JobInfo is deleted (all entries in the array). If a parameter is given, only that index in the JobInfo array is deleted.

hex to binary	Interprets the value of the parameter as a hexadecimal string and converts it to binary data.
jobinfo	The parameter value is passed through standard JobInfo Substitution and the result is a name of a JobInfo, the value of which is assigned to the chosen JobInfo.
prepend	Prepends (places in front of) the value of the parameter. The parameter value is passed through standard JobInfo Substitution before being prepended.
regex	Treats the value as a regular expression.
split	Converts the given JobInfo from a single value to an array of values. The value is split on whatever the parameter evaluates to, e.g. a semicolon.

10.19 JobInfo Scanner

The JobInfo Scanner scans for JobInfos in Text or XML as input format.

10.19.1 Text as input format

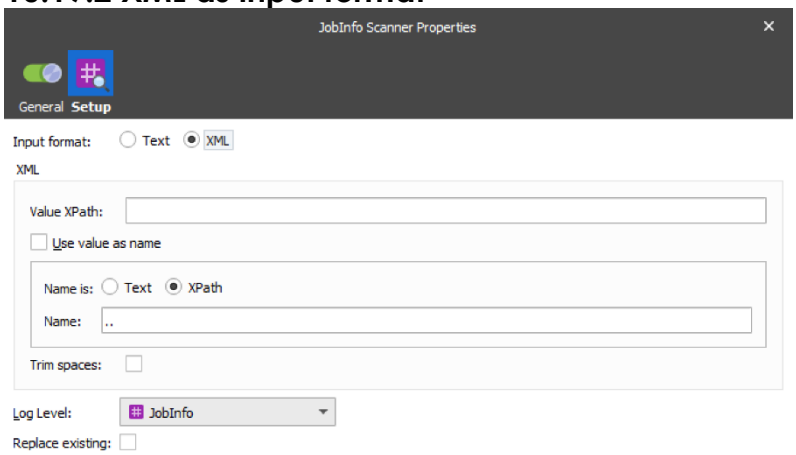


Scans text data for the syntax **`#JobInfo Name=Value#`** and works on 8-bit data only.

The table below provides a brief description of the action properties:

Property	Description
Extraction Mode	Specifies whether the embedded JobInfo Syntax in data must be Leaved, Overwritten or Removed after scanning of data.
Overwrite Character	Character in hexadecimal to overwrite the characters in the JobInfo syntax string.
Log level	To extract the JobInfos from the document and create them on the current job, you need to specify a category under which the events related to the JobInfo are logged. **JobInfo** The events related to the JobInfo are logged under the JobInfo category. **Debug** The events related to the JobInfo are logged under the Debug category. **NoLog** The events related to the JobInfo are not logged.

10.19.2 XML as input format



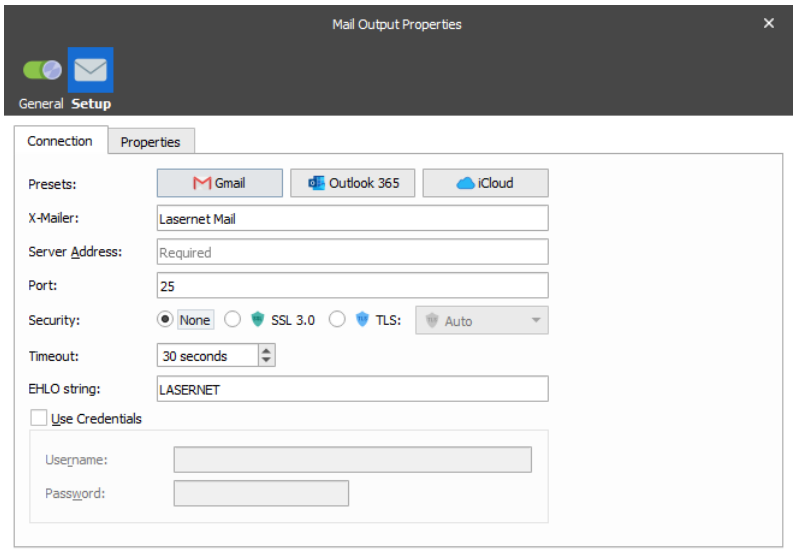
Scan XML data by defining an XPath for value and name. Works similar to the **Create JobInfo** action in the XML Transformer – see Lasernet XML Transformer manual.

The table below provides a brief description of the action properties:

	Property	Description
1	Value XPath	The XPath value specifies which nodes contain the values for JobInfos.
2	Name	Specify a JobInfo name.
3	Use value as name	Select the check box to apply the Value XPath as a JobInfo name.
4	Replace existing	Select the check box to replace any other JobInfo with the same name. Some JobInfos can contain a list of several values for example, an email distribution list.
5	Node name is	**Text** – select the radio button to interpret the value indicated in the **Name** field as text. **XPath** – select the radio button to interpret the value indicated in the **Name** field as an XPath.
6	Log level	To extract the JobInfos from the document and create them on the current job, you need to specify a category under which the events related to the JobInfo are logged. **JobInfo** The events related to the JobInfo are logged under the JobInfo category. **Debug** The events related to the JobInfo are logged under the Debug category. **NoLog** The events related to the JobInfo are not logged.

10.20 Mail Output

Used for sending a job as email via the SMTP protocol.



10.20.1 Settings

See section for Mail Output module for more information.

10.21 OAuth 2.0

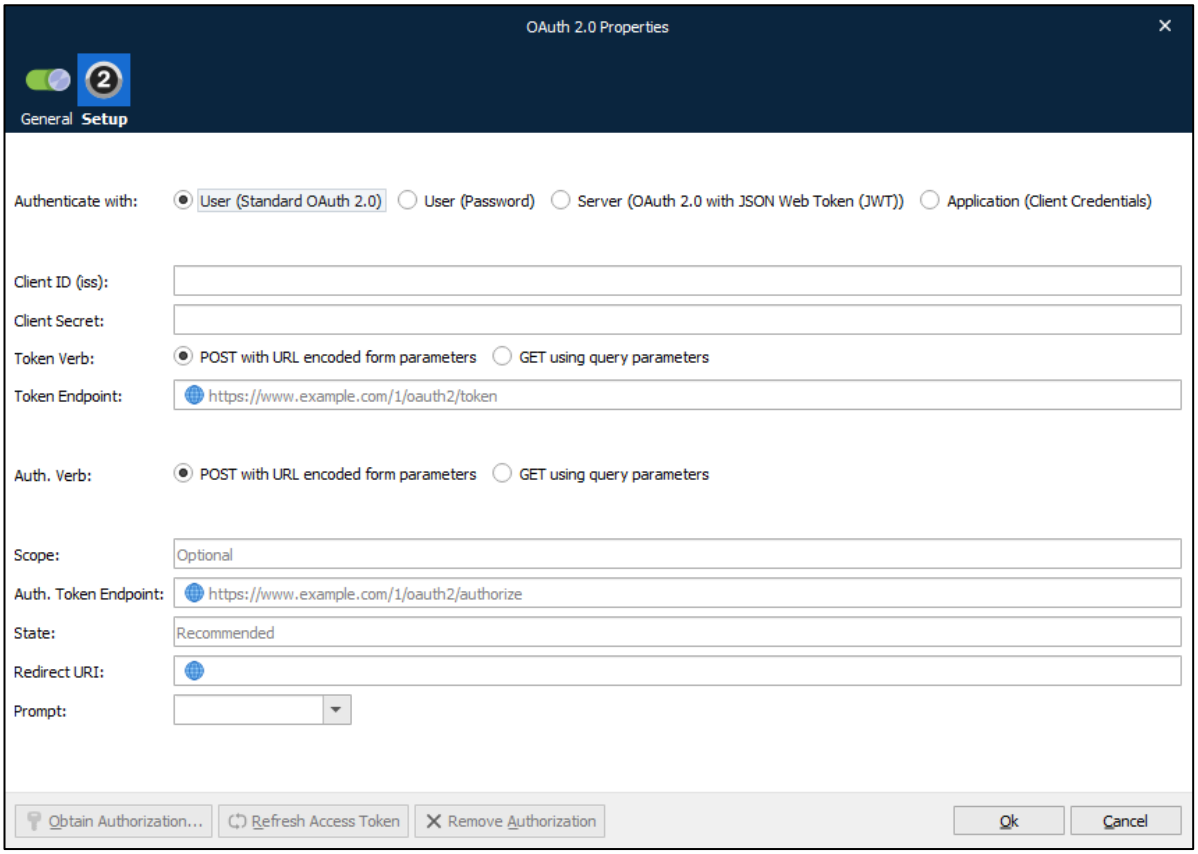
The OAuth 2.0 modifier is used for token-based security. It supports various flows, including a JSON Web Token (JWT) based flow. The modifier provides authorization for a given user or application identity and returns the access token. The access token is set as a JobInfo and available for HTTP requests in other modules.

You must create Lasernet as an app on the website of the service provider that you want to connect to. Then, you must configure the authentication and permissions to begin using the third-party API. All the settings values that you require for the OAuth 2.0 modifier are available from the setup of your third-party application.

10.21.1 User (Standard OAuth 2.0)

Standard OAuth 2.0 user authentication requires you to log in with a username and password to authorize the module to access content for the service account.

After you have entered appropriate values for the settings, click **Obtain Authorization**.



10.21.1.1 Settings

Client ID (iss) — Lasernet app credentials (ID) as registered with the service provider.

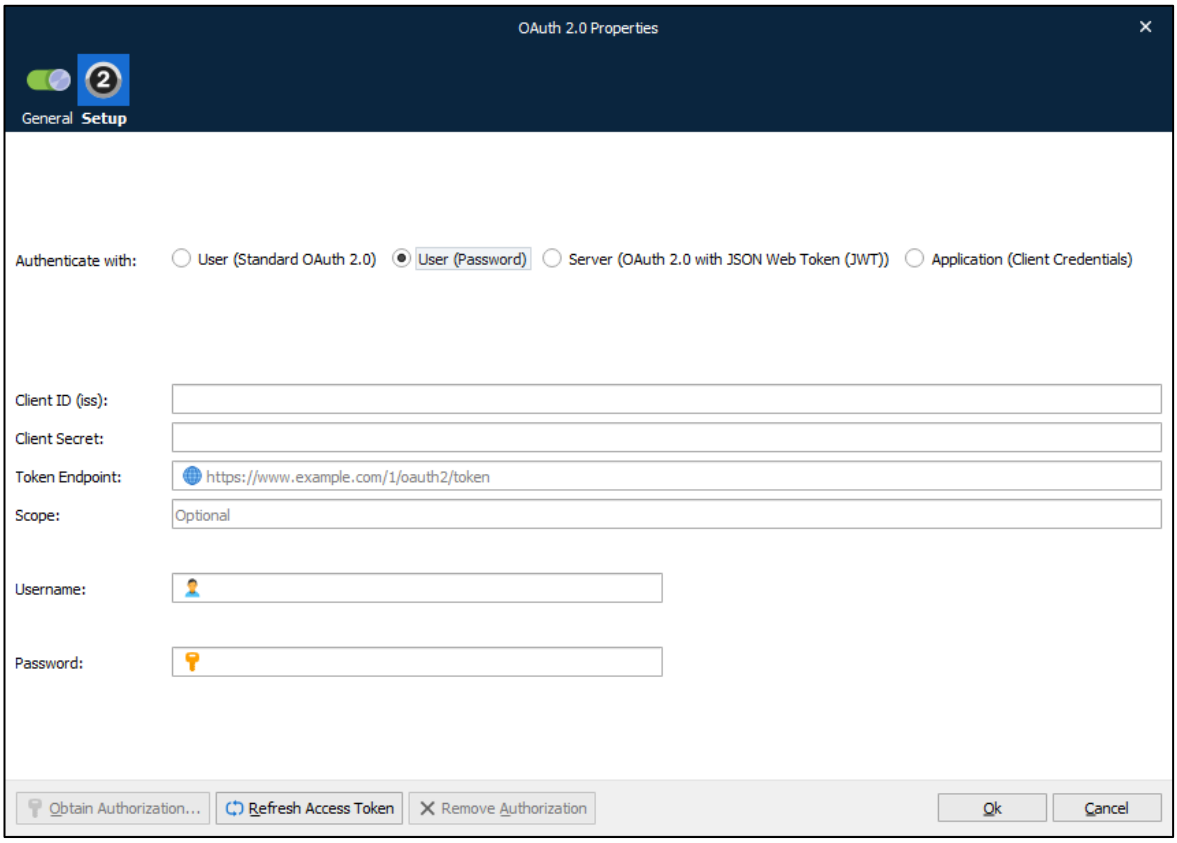
Client Secret — Lasernet app credentials (secret key) as registered with the service provider.

Token Verb	Select the appropriate verb for the request to the token endpoint. Supported options are **POST with URL encoded form parameters** and **GET using query parameters**. To understand which option to select, refer to the endpoint API.
Token Endpoint	The token endpoint URL of the authorization server is used as a value for an "aud" element to identify the authorization server.
Auth. Verb	Select the appropriate verb for the request to the authorization endpoint. Supported options are **POST with URL encoded form parameters** and **GET using query parameters**. To understand which option to select, refer to the endpoint API.
Auth. Token Endpoint	The authorization URL that you will redirect the user to.
Scope	The request might have one or more scope values indicating additional access requested by the application. The authorization server will need to display the requested scopes to the user. This setting is optional.
State	The **State** parameter is used by the application to store request-specific data, prevent CSRF attacks, or both. The authorization server must return the unmodified state value back to the application. Using this setting is optional but recommended.
Redirect URI	The redirect URI is the URL within your application that will receive OAuth 2.0 credentials. For example: `https://localhost`
Prompt	Specifies the type of interaction that the authorization service will have with the user when they are taken to the **Auth. Token Endpoint** URL. To understand the specific interaction that each valid option (**none**, **consent**, **select_account**, and **login**) results in, refer to the endpoint API.

10.21.2 User (Password)

This option configures the OAuth 2.0 modifier to use the OAuth 2.0 Password grant type.

After you have entered appropriate values for the settings, click **Ok**. Optionally, to test the credentials that you entered, click **Refresh Access Token** before you click **Ok**.



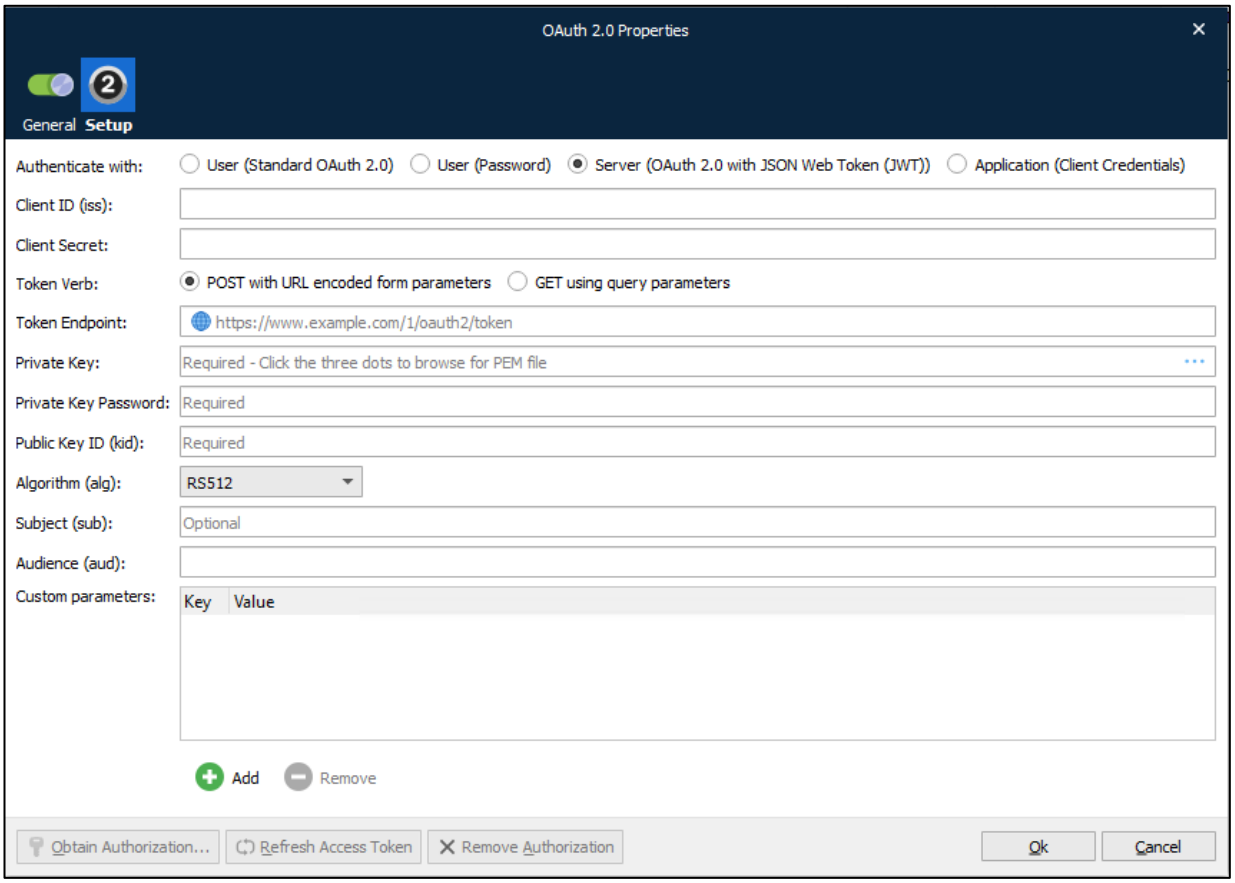
10.21.2.1 Settings

Client ID (iss)	Lasernet app credentials (ID) as registered with the service provider.
Client Secret	Lasernet app credentials (secret key) as registered with the service provider.
Token Endpoint	The token endpoint URL of the authorization server.
Scope	The request might have one or more scope values indicating additional access requested by the application. The authorization server will need to display the requested scopes to the user. This setting is optional.
Username	Part of the user credentials that Lasernet will exchange for an access token.
Password	Part of the user credentials that Lasernet will exchange for an access token.

10.21.3 Server (OAuth 2.0 with JSON Web Token (JWT))

OAuth 2.0 with JWT (Server Authentication) enables the module to authenticate directly to a service, using a digitally-signed JSON Web Token (JWT) instead of user credentials.

After you have entered appropriate values for the settings, click **Ok**. Optionally, to test the credentials that you entered, click **Refresh Access Token** before you click **Ok**.



10.21.3.1 Settings

Client ID (iss) — Lasernet app credentials (ID) as registered with the service provider.

Client Secret — Lasernet app credentials (secret key) as registered with the service provider.

Token Verb — Select the appropriate verb for the request to the token endpoint. Supported options are **POST with URL encoded form parameters** and **GET using query parameters**. To understand which option to select, refer to the endpoint API.

Token Endpoint (aud) — The token endpoint URL of the authorization server is used as a value for an "aud" element to identify the authorization server as an intended audience of the JWT.

Private Key — Generate an RSA keypair to sign and authenticate the JWT request made by your app. Download and save a copy of the private key (`.pem` file) in a Lasernet resource folder. Click **Browse** and then select the file.

Example of the format of a private key:

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIFDjBABgkqhkiG9w0BBQ0wMzAbBgkqhkiG9w0BBQwwDgQIN5gcZd3m0XwCAggA
MBQGCCqGSIb3DQMHBAiIEnj1jz73MASCBMjI3q8SDNKcEHpxywyv8tMwxHeovc2m
.
HYY=
-----END ENCRYPTED PRIVATE KEY-----
```

Private Key Password Password for the private key.

Public Key ID (kid) ID for the public key.

Algorithm (alg) Supported algorithms: RS256, RS384, RS512, ES256, ES384, ES512.

Subject (aud) The "sub" (subject) claim identifies the principal that is the subject of the JWT. The claims in a JWT are normally statements about the subject. The subject value must either be scoped to be locally unique in the context of the issuer or be globally unique. The processing of this claim is generally application-specific. The "sub" value is a case-sensitive string containing a StringOrURI value.

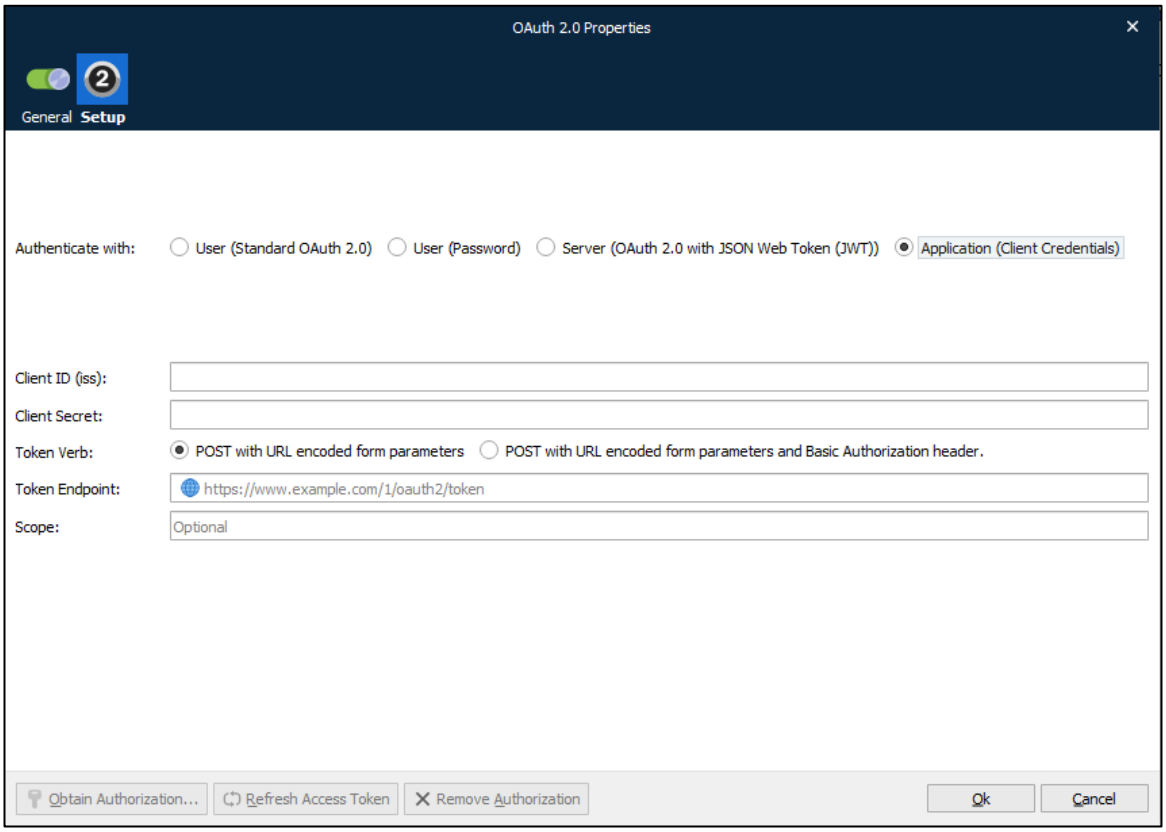
Audience (aud) The "aud" (audience) claim identifies the recipients that the JWT is intended for. Each principal intended to process the JWT must identify itself with a value in the audience claim. If the principal processing the claim does not identify itself with a value in the "aud" claim when this claim is present, then the JWT must be rejected. In the general case, the "aud" value is an array of case-sensitive strings, each containing a StringOrURI value. In the special case when the JWT has one audience, the "aud" value can be a single case-sensitive string containing a StringOrURI value. The interpretation of audience values is generally application-specific.

Customer parameters Add the list of customer parameters required by the service provider.

10.21.4 Application (Client Credentials)

This option configures the OAuth 2.0 modifier to use the OAuth 2.0 Client Credentials grant type.

After you have entered appropriate values for the settings, click **Ok**. Optionally, to test the credentials that you entered, click **Refresh Access Token** before you click **Ok**.



10.21.4.1 Settings

Client ID (iss) — Lasernet app credentials (ID) as registered with the service provider.

Client Secret — Lasernet app credentials (secret key) as registered with the service provider.

Token Verb — Select the appropriate verb for the request to the token endpoint. Supported options are **POST with URL encoded form parameters** and **POST with URL encoded form parameters and Basic Authorization header**. To understand which option to select, refer to the endpoint API.

Token Endpoint — The token endpoint URL of the authorization server.

Scope — The request may have one or more scope values indicating additional access requested by the application. The authorization server will need to display the requested scopes to the user. This setting is optional.

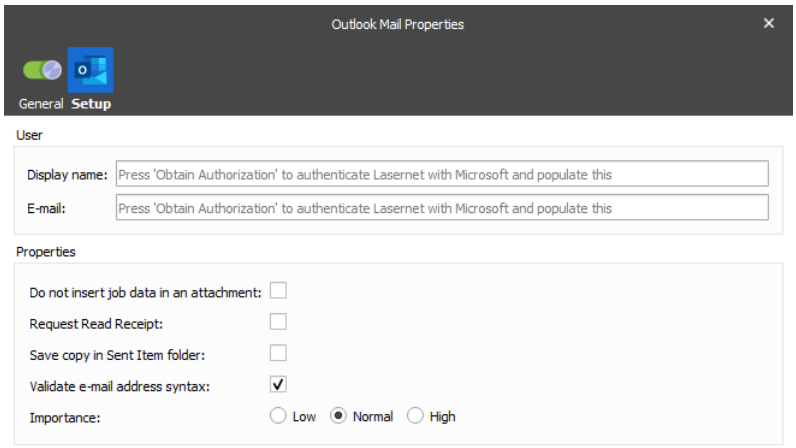
10.21.5 JobInfos

The OAuth 2.0 modifier sets a JobInfo.

OAuth2AccessToken	Value of the OAuth 2.0 access token.

10.22 Outlook Mail

The Outlook Mail module integrates directly with Office 365 to send emails.

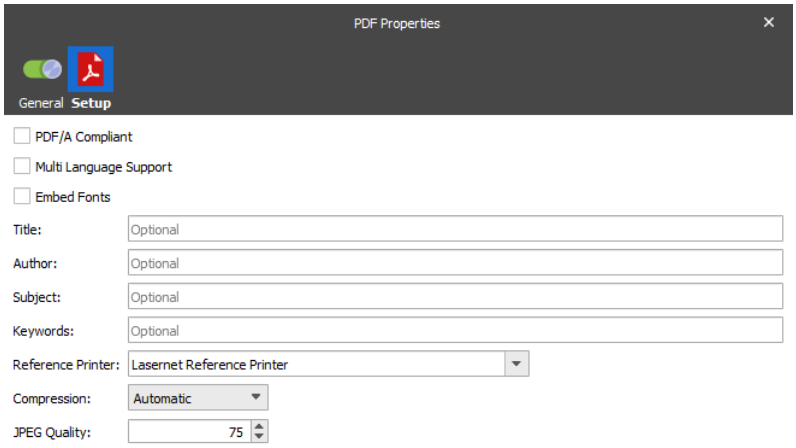


10.22.1 Settings

See section for Outlook Mail output module for more information.

10.23 PDF

The PDF modifier converts EMF data, created by the Lasernet EMF printer driver, the Overlay Engine or the Form Engine, to PDF (reference version 1.4). The PDF modifier will set the paper size and orientation for the document based on the paper formats included in EMF data.



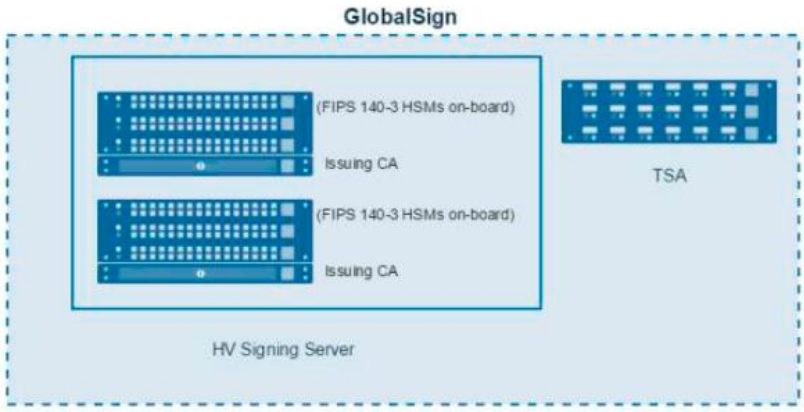
10.23.1 Settings

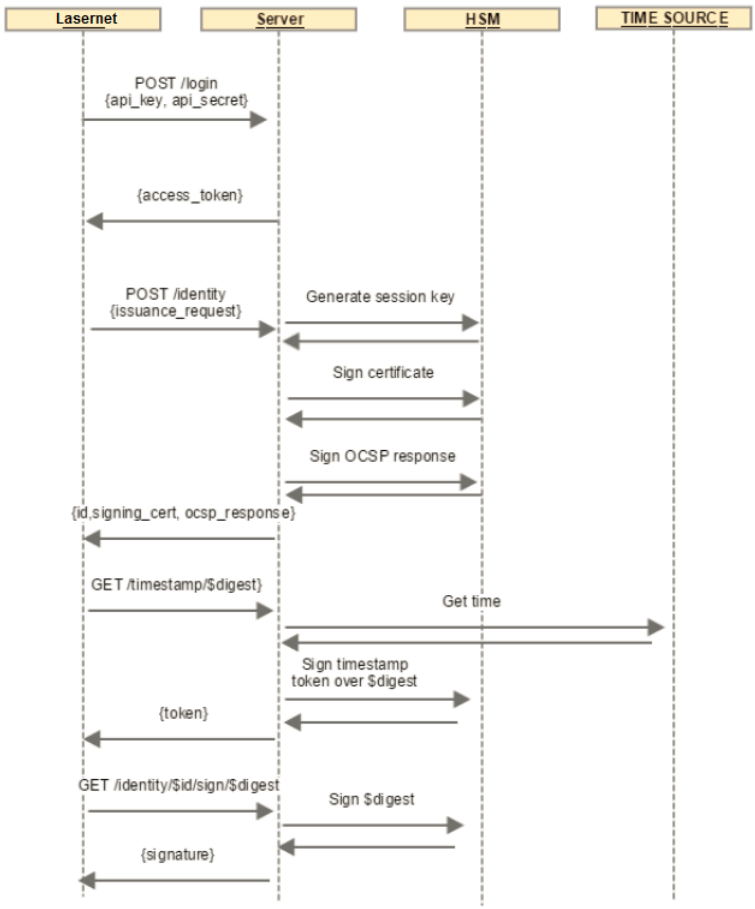
See section for PDF Engine for more information.

10.24 PDF Cloud Security

GlobalSign offers cloud-based document signing. Please contact your GlobalSign service provider to obtain the required API Key, API Secret, Client Certificate and password

Each user will need to create an account, (handled by the service provider) and use the assigned issuance rate, quota and type of certificates.





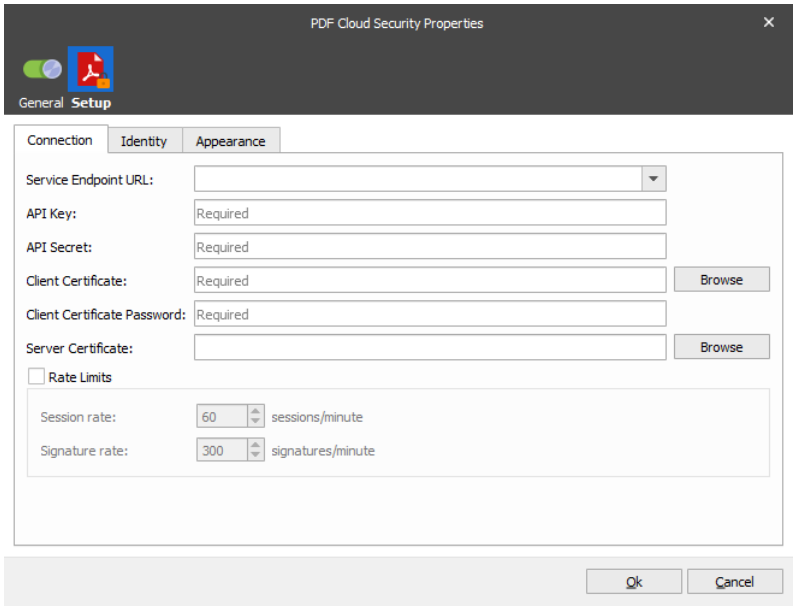
Once an account is created, authentication information will be sent to you and must be included in the login and initiate signing sessions of this modifier.

The process to establish this is as follows: Send your provider your public key from the created key pair.

GlobalSign creates a PEM encoded x509 certificate based on that key that has client authentication enabled and is used to connect to the service.

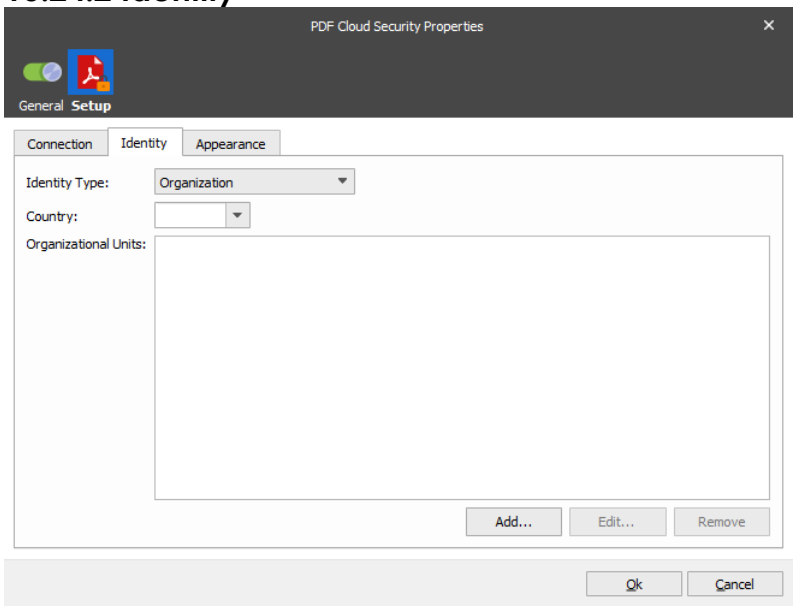
The modifier will use this to establish an mTLS connection to the service.

10.24.1 Connection



Initiate signing session: This establishes the secure login connection using the API Authentication provided.

10.24.2 Identity

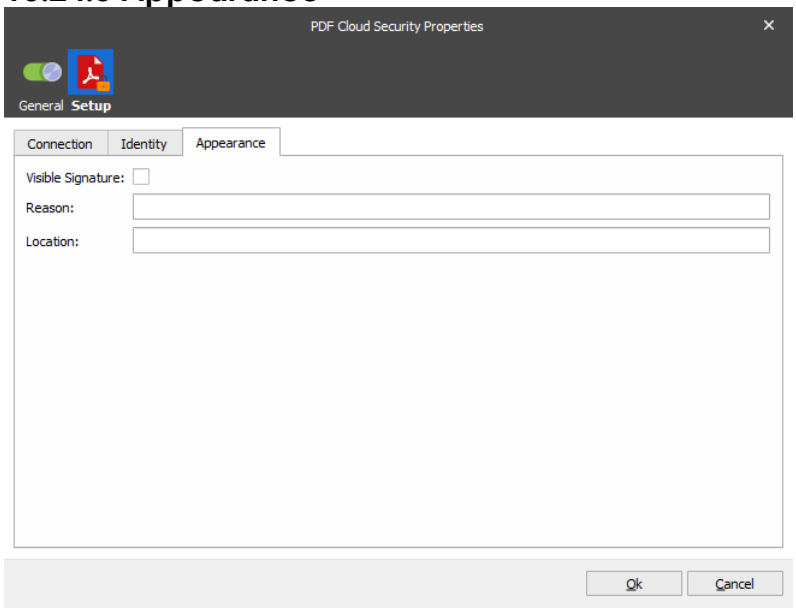


Create a signing certificate: this must be done before performing a signing operation.

A private key is created per signing session if no existing key is active, or the previous key has expired (signing sessions expire automatically in <15 minutes). A new key is also created if you need to sign in using a different identity.

The signees' identity information is specified here to create the signing certificate.

10.24.3 Appearance



Set your reason and location for signing the PDF document.

10.25 PDF Extract

This modifier is used to extract metadata and files embedded in the PDF/A-3 format.

10.25.1 Settings

There are no user settings available for this modifier.

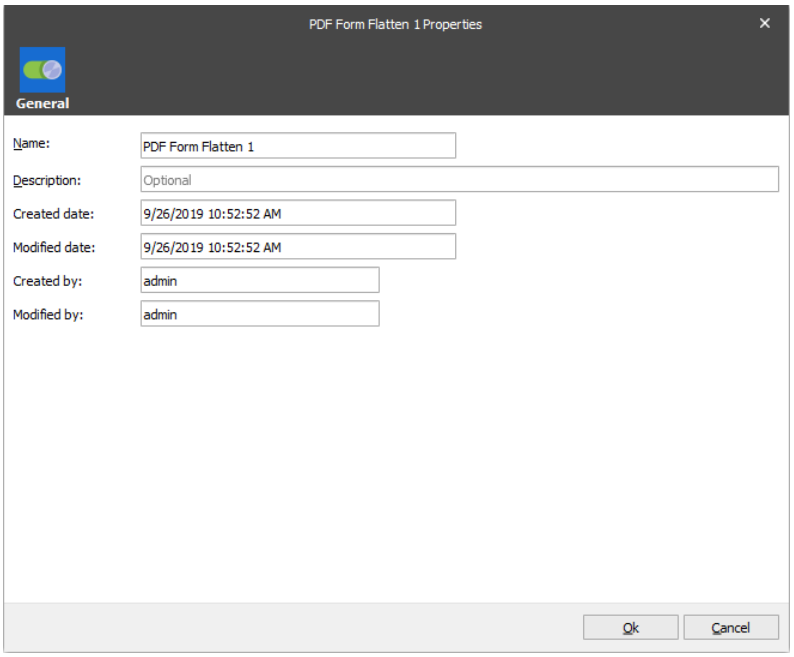
10.25.2 JobInfos

The PDF Extract module sets the following JobInfos

PDFAuthor	Content (if any) of the Author field in the PDF.
PDFCreator	Content (if any) of the Creator field in the PDF.
PDFKeywords	Content (if any) of the Keywords field in the PDF.
PDFSubject	Content (if any) of the Subject field the PDF.
PDFTitle	Content (if any) of the Title field in the PDF.
PDFCompany	Content (if any) of the Company field in the PDF.
PDFProducer	Content (if any) of the Producer field in the PDF.
PDFCreationDate	The creation date of the PDF.
PDFModificationDate	The modification date of the PDF.
PDFEmbedDescription	Plaintext description of the embedded file (array).
PDFEmbedSubtype	MIME type of the embedded file. Defaults to application/octet-stream (array).
PDFEmbedData	Contents of the embedded file (array).
PDFEmbedCreateDate	The creation date of the embedded file (array).
PDFEmbedModDate	The modification date of the embedded file (array).
PDFMetadata	Content (if any) of the XMP metadata block of the PDF.

10.26 PDF Form Flatten

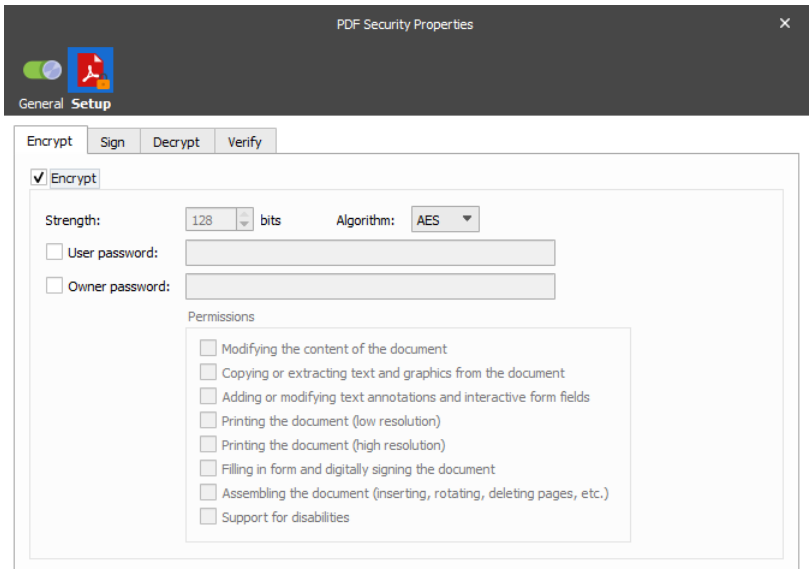
This modifier is used to flatten a PDF document containing interactive elements such as text boxes, check boxes, combo boxes, etc. and lock it for changing values of such fields.



When you use the forms to add values to interactive fields in a PDF document and make this document either flatten or non-flatten to have different versions of it for sending them to different destinations, it is possible to apply this modifier on a specific stage of the flow by defining a corresponding criterion.

10.27 PDF Security

This modifier is used for adding support for security options (change content, printing, copying content, adding notes or modifying form fields, signing, etc.).



10.27.1 Encrypt

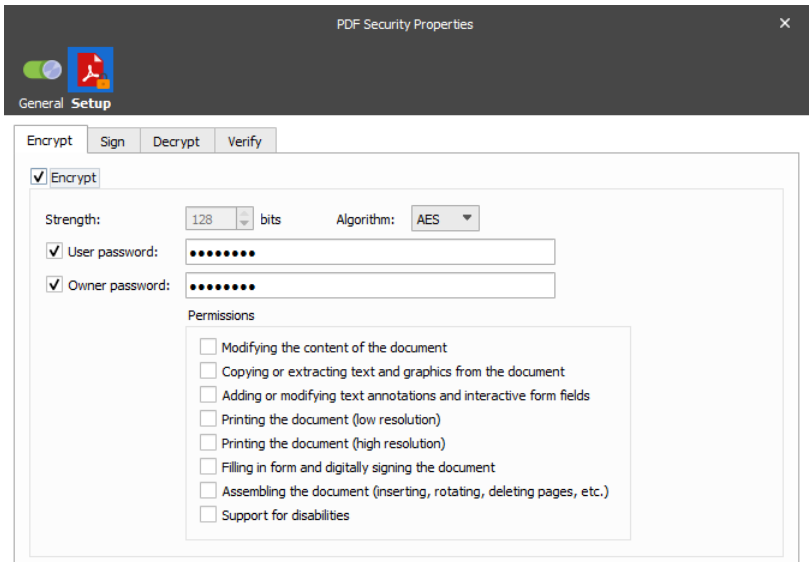
PDF encryption allows you to secure the contents of your PDF with a password. It also allows you to define which operations will be allowed by the user reading the PDF.

The user and owner password created for the PDF document can be overwritten at runtime by setting the JobInfos:

PDFOwnerPassword

PDFUserPassword

Please note that encryption is not allowed when working with PDF/A (ISO 19005-1-2005).



Algorithm choice

The following algorithms can be used for encrypting the PDF. Each algorithm has its own advantages and disadvantages which should be taken into consideration when deciding which one to use.

RC4

The RC4 algorithm is supported from PDF 1.1 (Acrobat Reader 2.0). It supports key lengths ranging from 40 bits to 128 bits in 8 bit increments. Use of key lengths longer than 40 bits is supported from PDF 1.4 (Acrobat Reader 5.0).

Be aware that the RC4 algorithm has been proven to have vulnerabilities that make it less secure than the AES algorithm. It should only be used for backward compatibility with earlier versions of Acrobat Reader.

AES

The AES algorithm is a newer than RC4 and provides better security. AES encryption was introduced in PDF 1.6 (Acrobat Reader 7.0) and only supports a key length of 128 bits. This is the recommended algorithm when you do not require backward compatibility with older versions of Acrobat Reader.

10.27.2 Passwords

Two passwords are defined when enabling encryption. The user password is the password that must be supplied to the recipient of the PDF. When opening the PDF using the user password, the restrictions defined by the document permissions will be in effect.

Opening the PDF using the owner password will override the defined restrictions. You will also be able to change the operations allowed by the recipient.

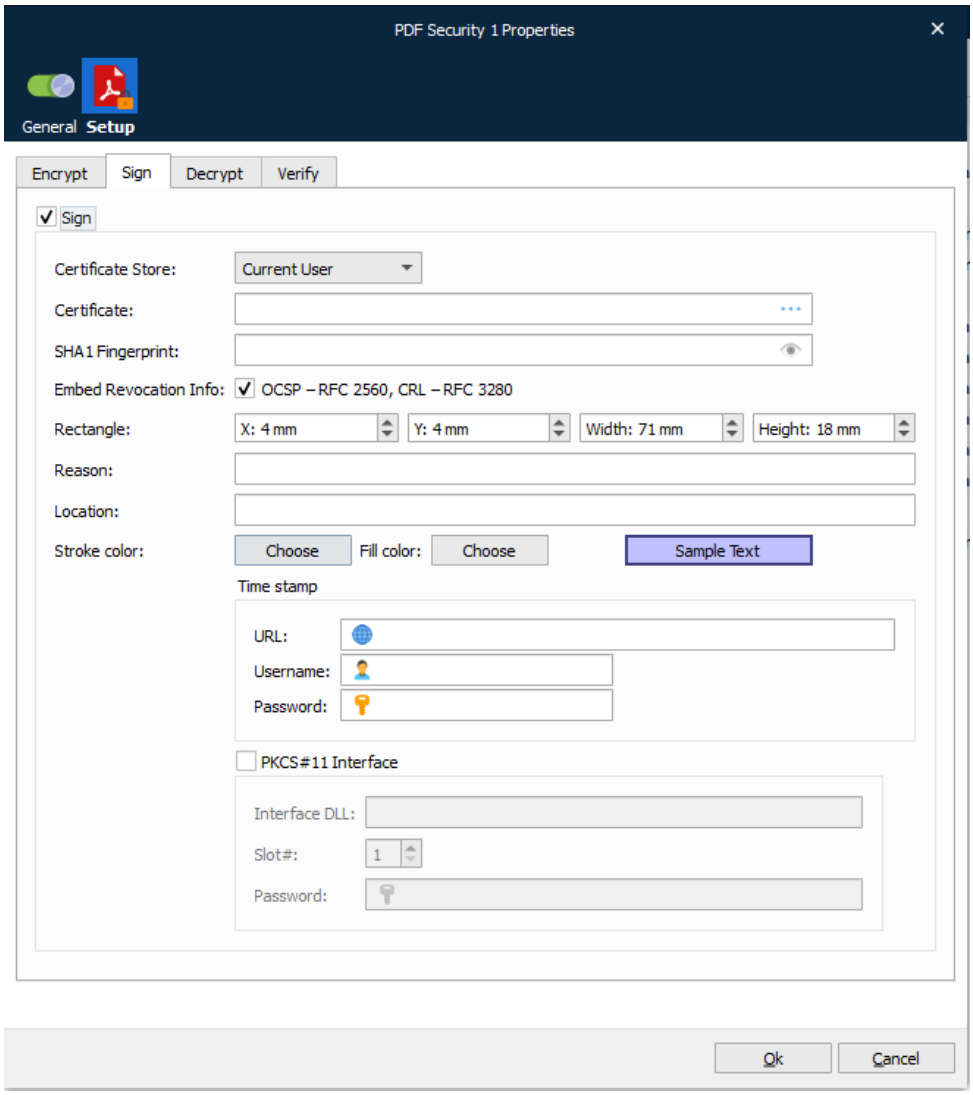
If the user password and the owner password are the same, the user password will have highest priority. This means you will not be able to circumvent the restrictions defined by the document permissions.

10.27.3 Permissions

You can define which actions will be allowed when opening the encrypted PDF using the user password. This allows you to prevent the document from being modified, printed or copied from.

10.27.4 Sign

By signing your PDF using a digital certificate, you allow the recipient to verify both the origin and the integrity of the document. This means that the recipient is able to verify that the document came from your organization and has not been changed since it was created. Digital signatures are supported since PDF 1.3 (Acrobat Reader 4.0) and are allowed in PDF/A.



10.27.4.1 Certificate Store

Select whether the certificate is located on the Local Machine or for the Current User account.

10.27.4.2 Certificate

In order to sign the PDF using a digital signature, you must have a valid and appropriate certificate installed for the user account that the Lasernet service is running under.

10.27.4.3 SHA1 Fingerprint
The SHA1 fingerprint is used to verify that the signed file is unaltered. The checksum is created before the file is transmitted, and then once again it reaches its destination.

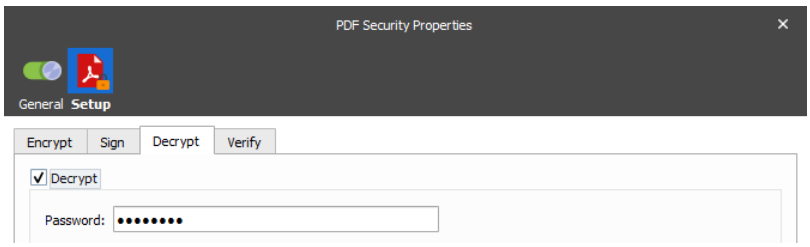
10.27.4.4 Rectangle (X, Y, Width, Height)
Specify the location (**X** and **Y**) and size (**Width** and **Height**) of the overlay that this modifier adds to the document.

10.27.4.5 Reason
The Reason field allows you to enter a text string that will be displayed when viewing the signed PDF. This provides a visual representation of the signature to the user. You can select text and background colour for the text field.

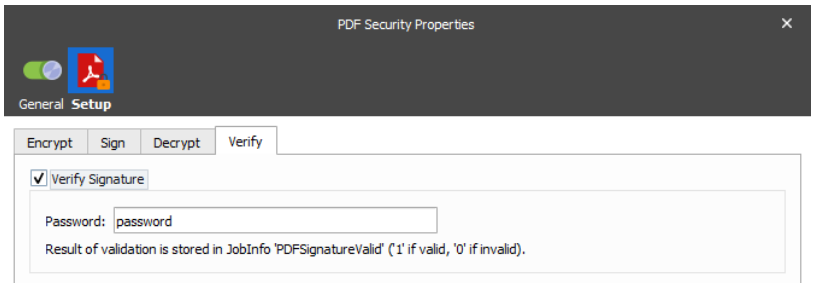
10.27.4.6 Location
The physical location where the signature is added for example “Zurich, Switzerland”. If this property is set to an empty string no entry is created.

10.27.4.7 Timestamp
By signing your PDF with a timestamp provided by a trusted source, you can make sure that the PDF will still show a valid signature after your certificate expires. If you choose not to timestamp your PDF, the recipient will receive a warning that the certificate used to sign the PDF has expired.

10.27.5 Decrypt
The PDF Security modifier allows you to remove password encryption from an already encrypted PDF. This may be required in order to further process the PDF. In order to successfully remove encryption from a PDF you must supply the correct password.



10.27.6 Verify
The PDF Security modifier allows you to verify the signature of an incoming PDF. You can supply a decryption password in case the incoming PDF is encrypted. The result of the validation will be stored in the **PDFSignatureValid** JobInfo.



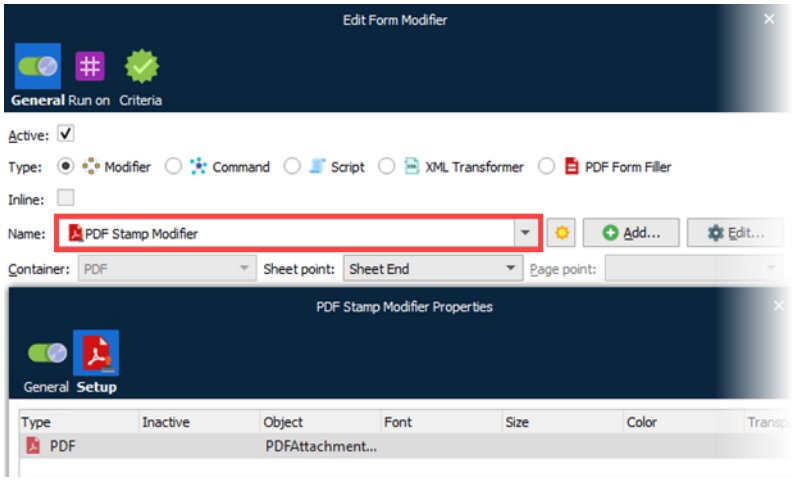
10.28 PDF Stamp

You can apply one or more text strings and images as stamps to pages in a PDF document.

Images must be loaded into a JobInfo, before the modifier is executed, as binary graphics in PNG, JPEG or TIFF format.

Notes:

- The sheet design must be processed before using the PDF Stamp Modifier.
- The modifier can be applied at sheet or module level.

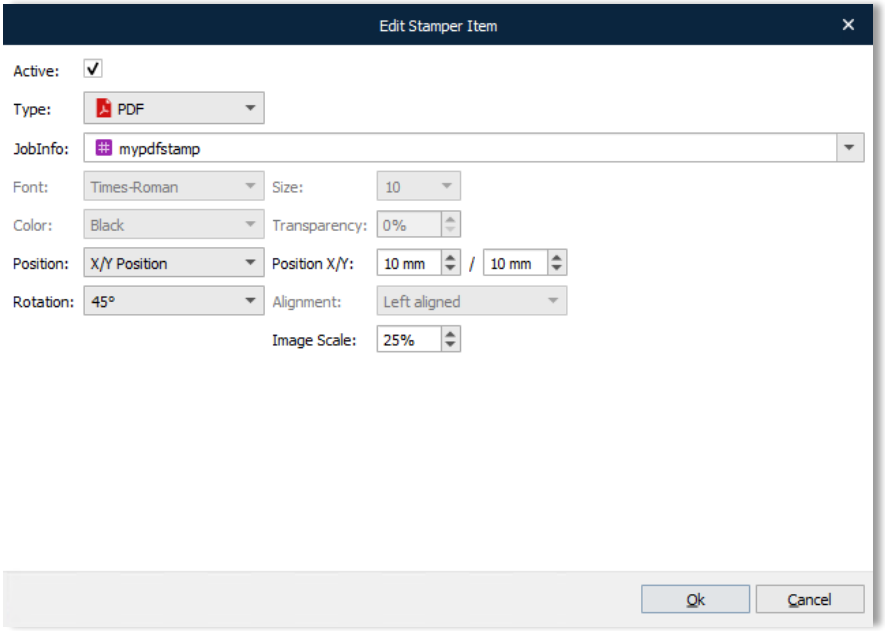


Click **Add** to configure a new stamper item.

Click **Edit** to edit an existing stamper item.

The modifier can contain a list of items, where the items are inserted as layers in the PDF document, in the specified order.

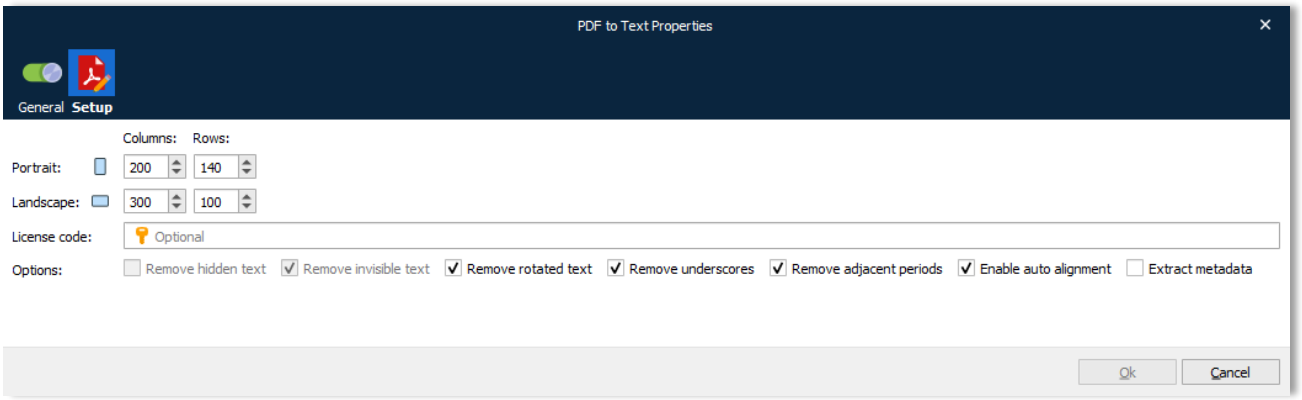
10.28.1 Settings



Type	Select Text or Image formats.
Text & Image JobInfo	A fixed text string with or without parameters for inserting page numbers. Type %P for current page and %T for total number of pages, like: “Page %P of %T”. Insert the contents of a JobInfo with binary graphics. Supported graphic formats: PNG, JPEG and TIFF.
Font	Times, Helvetica, Courier, Symbol and ZapfDingbats.
Size	Font sizes available: 4-12, 14, 16, 18, 20, 22, 24, 26, 28, 36, 48, 72 & 96 pt.
Color	Available values: Black, White, Red, Green and Blue
Transparency	Set a transparency value for the text where 0 % is opaque (darkest) and 100 % is fully transparent. Recommended for text inserted in the background.
Position & Margin X/Y	Set the position of the text. The value of the X/Y Position will insert the text in a position relative to the upper left corner of the paper. Top, Center and Bottom positions are relative to the X/Y margins.
Rotation	Rotate the text by the following degrees: 0, 45, 90, 135, 180, 225, 270, 315
Image Scale	Scale the image. Values below 100% will reduce the image size and higher than 100% will enlarge the image (compared to the original).

10.29 PDF to Text

All the pages in a PDF document are converted to text format as output with a FormFeed character (HEX 0C) as a splitter character between the pages.



The parameters to define the output text settings are:

Portrait Columns (Default=200). Recommended value for PDF documents, in Letter or A4 portrait format, is **200 columns**.

Portrait Rows (Default=140). Recommended value for PDF documents, in Letter or A4 portrait format, is **140 rows**.

Landscape Columns (Default=300). Recommended value for PDF documents, in Letter or A4 landscape format, is **300 columns**.

Landscape Rows (Default=100). Recommended value for PDF documents, in Letter or A4 landscape format, is **100 rows**.

License code: A license code is only needed, for users running Lasernet 6.7.1 and older, to run the module in compatibility mode. The latest and most optimized algorithm for converting PDF to Text is running in latest versions and does not require an additional 3rd party license.

Remove hidden text: Activate to remove hidden text in the PDF file, from appearing in output text. Only active for older versions.

Remove invisible text: Activate to remove invisible text, from OCR scanned the PDF documents, from appearing in the text output. Only active for older versions.

Remove rotated Text: Activate to remove characters rotated more than 5 % in PDF input from appearing in text output.

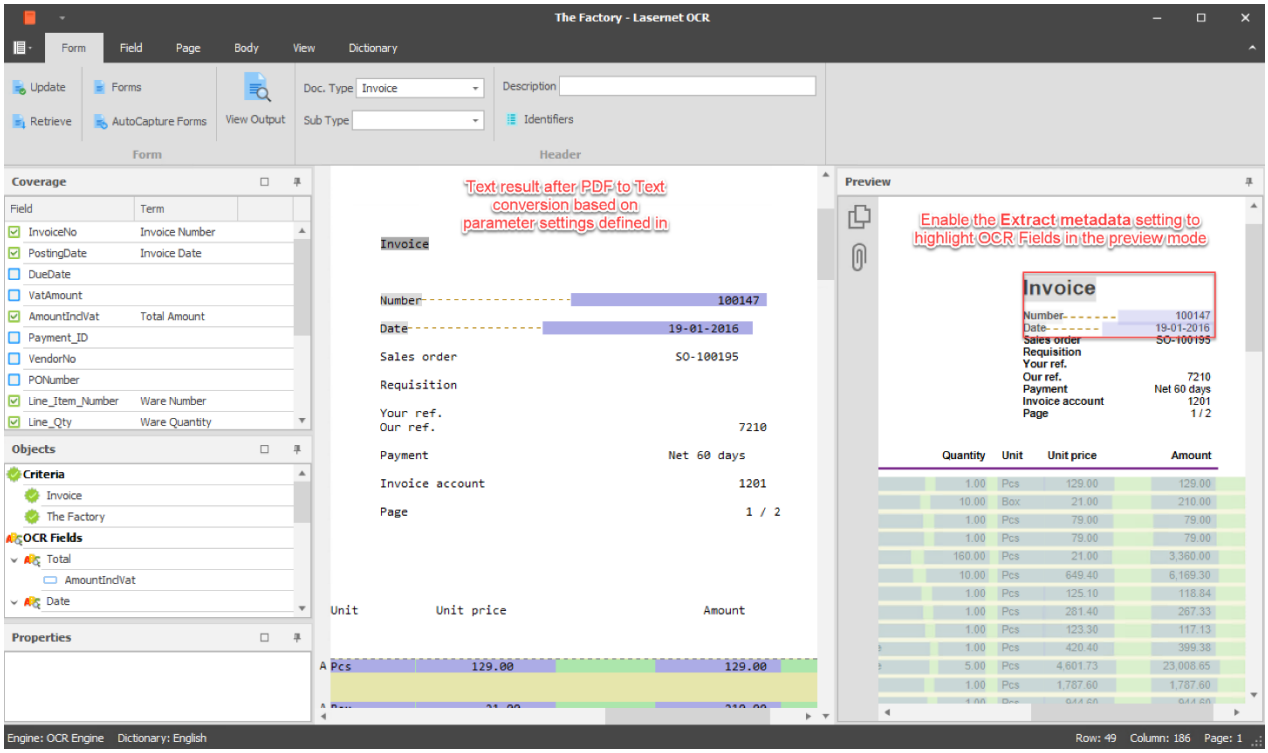
Remove underscores: Activate to remove underscores (HEX 5F) from appearing in text output.

Remove adjacent periods: Activate to remove adjacent (HEX 2E) from appearing in text output. It might be expected that a single period character will be left back after removing adjacent periods in a string.

Enable auto alignment: Activate to enable auto alignment of scanned text to ensure best output quality.

Extract metadata: Activate to highlight OCR Fields in the Lasernet OCR Editor → Preview window. This feature is memory consuming and will reduce speed dramatically for multi-pages documents.

Not recommended in a production environment with high volumes of incoming documents and for documents with more than 100 pages.



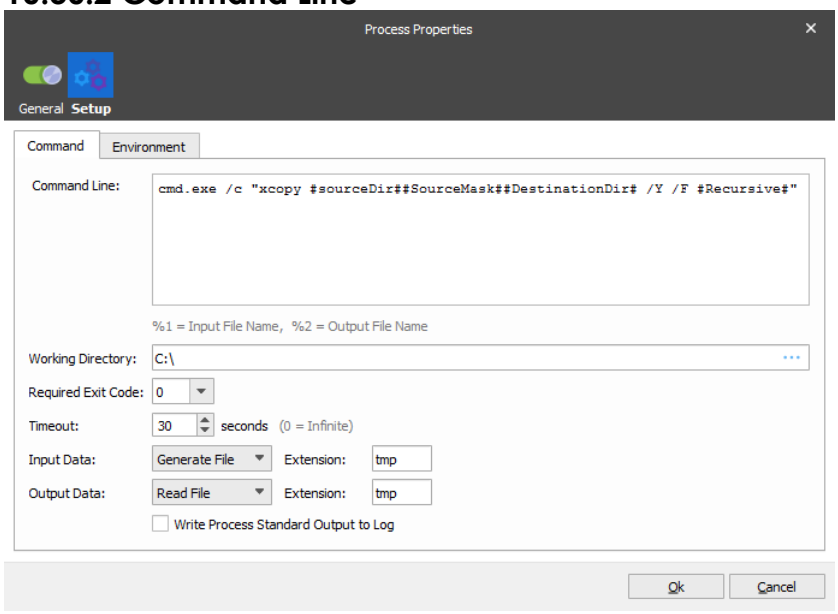
10.30 Process

The Process Modifier is used for running external processes. These processes can have data from Lasernet as input and produce output that is entered back into Lasernet. It is also possible to just run the external process and ignore any data transfer.

10.30.1 Settings

There are a variety of settings to control the Process Modifier. They are divided into Command Line, Process Settings and Environment.

10.30.2 Command Line



The above example calls the DOS xcopy command to copy some files. As can be seen, JobInfo substitution is available for the command line.

10.30.3 Process Settings

Working Directory

Sets the current directory for the external application. Can be left empty. Can include any number of #JobInfo# references for substitution.

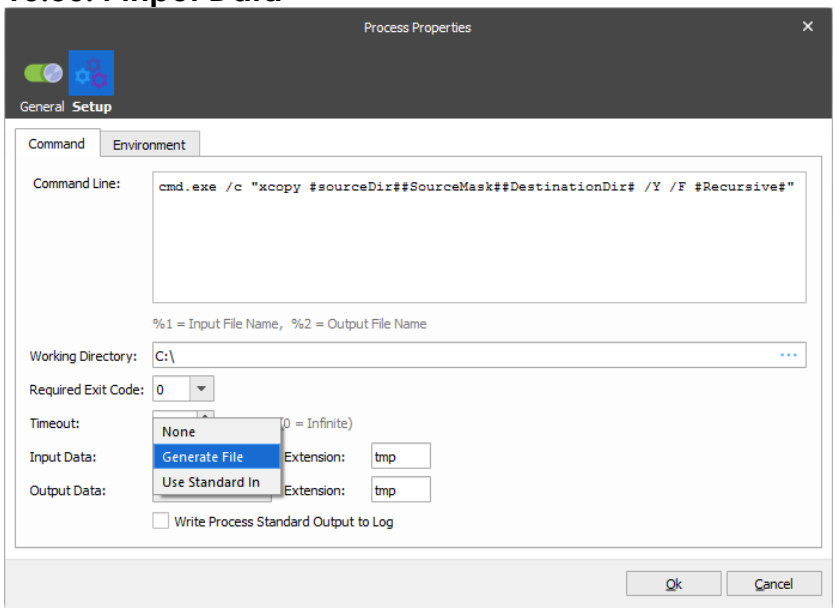
Timeout

How Lasernet should wait for the external application to finish. 0 (zero) means that it will wait indefinitely. This setting should be used with care. If the application does not finish, Lasernet will appear hung. If, on the other hand, the time out is too short, Lasernet will not be able to get the data returned from the application before it continues. Lasernet sets the JobInfo *ProcessTime* with the approximate number of seconds that the process ran for.

Required Exit Code

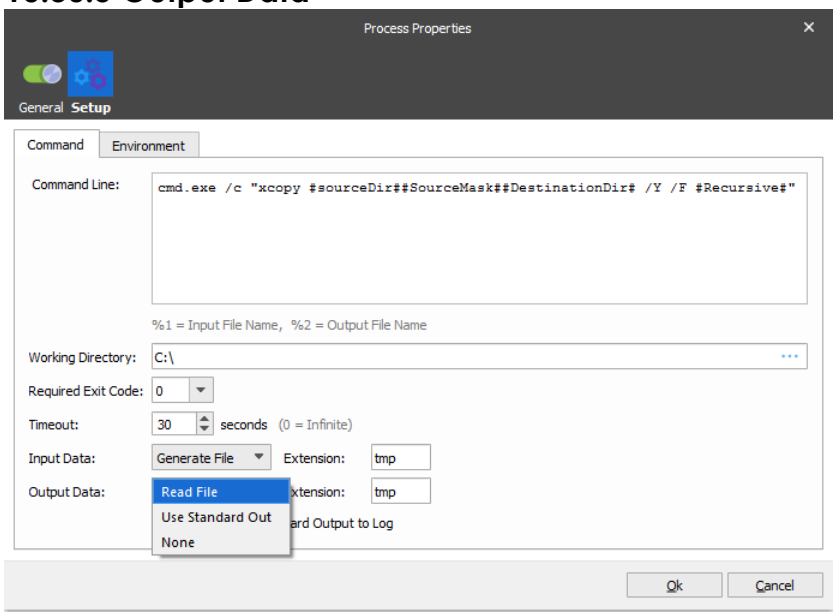
Which exit code to accept from the application as valid. *Ignore* means that it is not relevant. Lasernet will set the JobInfo *ExitCode* to the actual exit code.

10.30.4 Input Data



Determines how Lasernet should deliver data to the application. *Generate File* means that Lasernet creates a temporary file with the given *Extension* (the default is '*tmp'*). The filename generated is sent to the application with the *%1* setting on the *Command Line*. Lasernet sets the JobInfo *InTempFilename* with the filename. *Use Standard In* means that Lasernet sends the data to the application via its *stdin* pipe. *None* means that Lasernet should not send any data to the application.

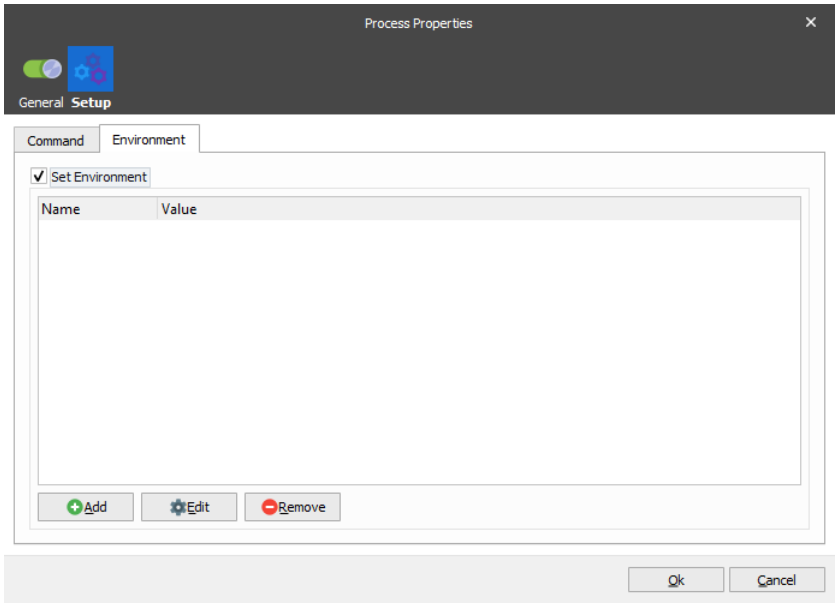
10.30.5 Output Data



Determines how Lasernet reads data created by the external application. *Read File* means that Lasernet sends a temporary filename to the application via the *%2* setting on the *Command Line*. Lasernet sets the JobInfo *OutTempFilename* with the filename. *Use Standard Out* means that Lasernet reads the data that the application sends to its *stdout* pipe. *None* means that Lasernet should ignore any data sent from the application.

When reading via *stdout* Lasernet sets a JobInfo *StdOut* to contain any data that was read that way.

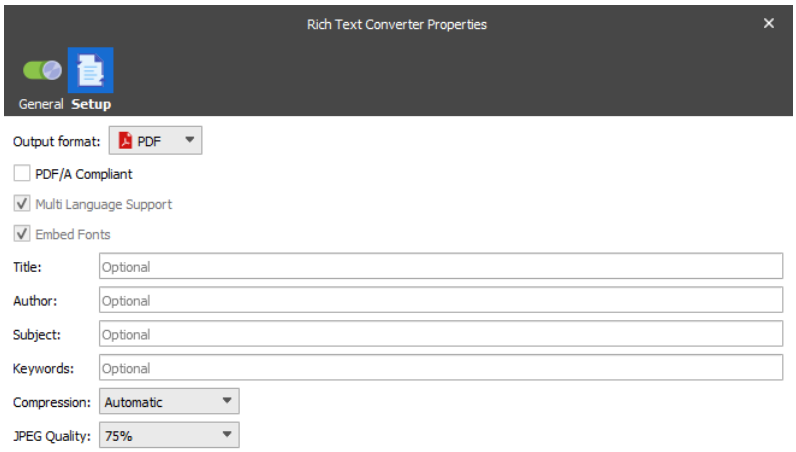
10.30.6 Environment



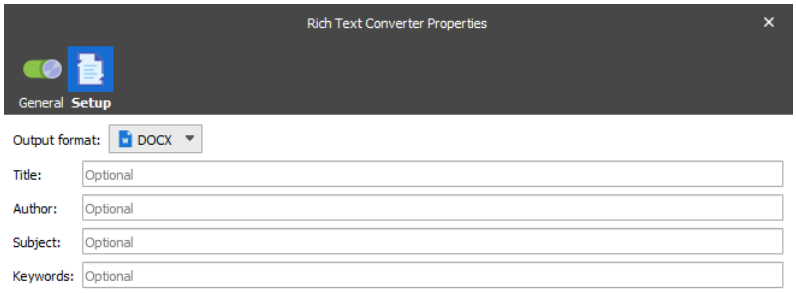
Normally the application inherits the environment from Lasernet, but in some cases you want to be able to set the environment yourself. This is possible on the Environment page. Each environment value supports JobInfo substitution.

10.31 Rich Text Converter

Select PDF as the output format to convert a document, created in the DOCX format, to a PDF or PDF/A compliant document.



Select **DOCX** to add additional phrases to a DOCX document.

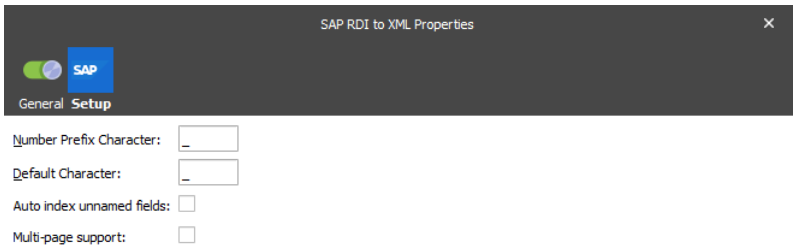


10.31.1 Settings

See section for Rich Text Convert Engine for more information.

10.32 SAP RDI to XML

Converts SAP RDI output to XML.

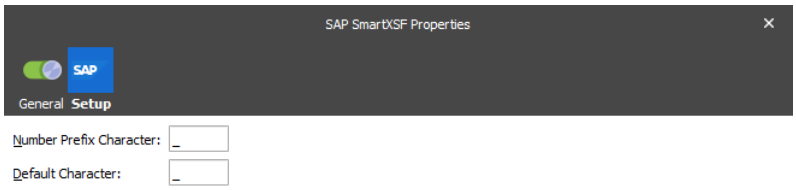


10.32.1 Settings

Number Prefix Character	If the SAP RDI tag begins with digits the modifier replaces it with this character as XML tags cannot begin with digits.
Default Character	If the SAP RDI tag contains characters that are not valid in an XML tag these characters are replaced with this *Default character.*
Auto index unnamed fields	If this field is activated, Lasernet will automatically include an auto index in the XML tags for unnamed fields in the SAP RDI file.

10.33 SAP SmartXSF Modifier

This module is intended for SAP XSF files. The output format is optimized for more accurate XML.



10.34 Tesseract-OCR

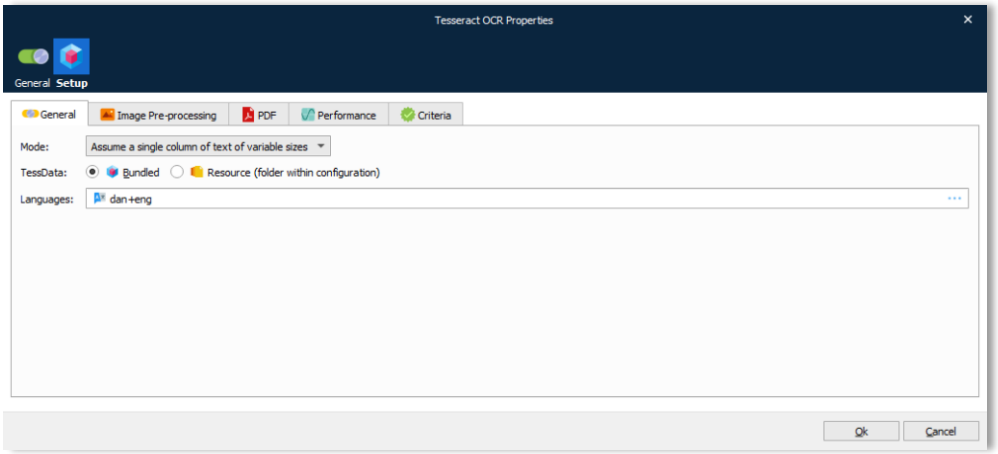
This module is based on Tesseract text recognition (OCR) engine, but improved in several ways by allowing PDF input, adding image pre-processing features, multi-threaded and outputting PDFs retaining original text from the input PDF.

10.34.1 General

The following input formats are valid:

- PDF
- TIFF
- PNG
- JPEG

Output is always a PDF where recognized text is added as a separate layer.



Mode

"Assume a single column of text of variable sizes" is recommended for invoice and order documents, with table structures, rare characters like EUR and USD sign and numbers that can't be verified against a language dictionary.

"Fully automatic page segmentation, but no OSD" is recommended for book-type text.

TessData

Tesseract uses AI to recognize text in images using trained data. Hand-written text, right-to-left and Asian languages are not supported. Select one of the following options:

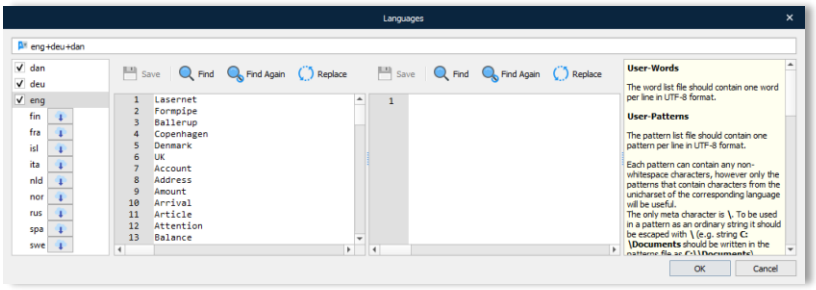
Bundled:

The module is trained and bundled with the following language packages: Danish, German, English, Finnish, French, Icelandic, Italian, Netherland, Norwegian, Russian, Spanish and Swedish.

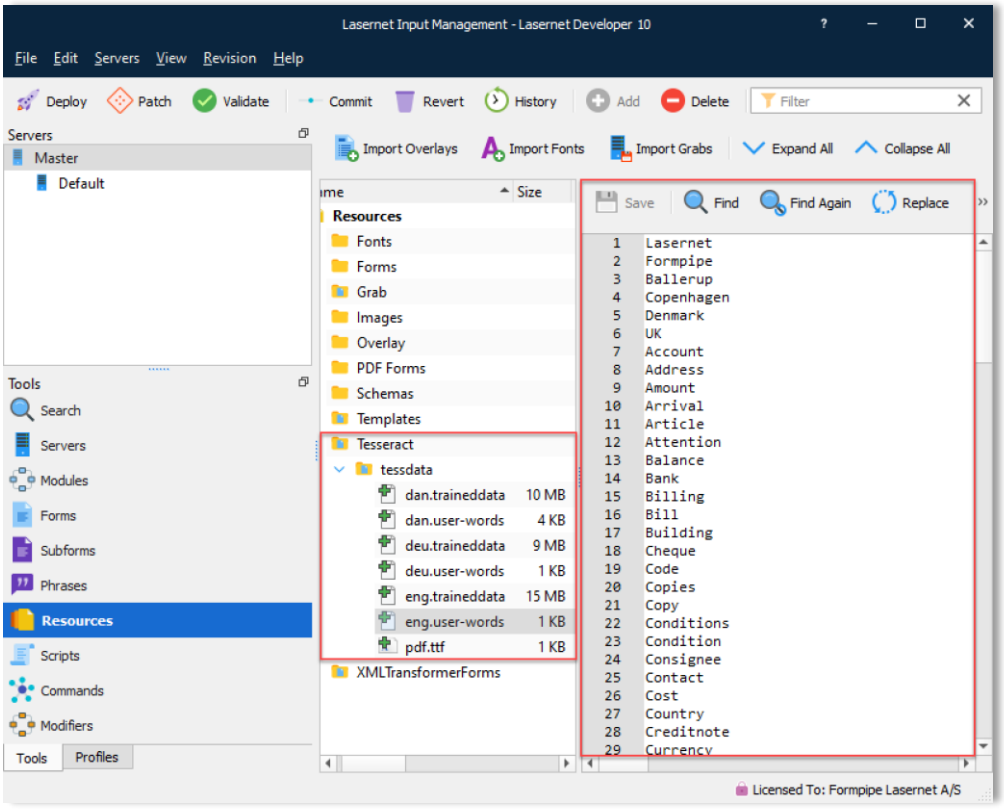
Resource:

Activate **Resource (folder within configuration)**

Click the  button to create a local copy of selected language packages and they will be stored in the Resources → Tesseract → tessdata folder of the configuration.



Go to Resources to maintain the list of user words and patterns.



User-Words: The word list file should contain one word per line in UTF-8 format.

User-Patterns: The pattern list file should contain one pattern per line in UTF-8 format.

Each pattern can contain any non-whitespace characters; however, only the patterns that contain characters from the unicharset of the corresponding language will be useful.

The only meta character is ****. To be used in a pattern as an ordinary string, it should be escaped with \ (e.g. string **C:\Documents** should be written in the patterns file as **C:\\Documents**).

This function supports a very limited regular expression syntax. One can express a character, a certain character class and a number of times the entity should be repeated in the pattern.

To denote a character class, use one of:

\c - unichar for which UNICHARSET::get_isalpha() is true (character)

\d - unichar for which UNICHARSET::get_isdigit() is true

\n - unichar for which UNICHARSET::get_isdigit() and UNICHARSET::isalpha() are true

\p - unichar for which UNICHARSET::get_ispunct() is true

\a - unichar for which UNICHARSET::get_islower() is true

\A - unichar for which UNICHARSET::get_isupper() is true

***** could be specified after each character or pattern to indicate that the character/pattern can be repeated any number of times before the next character/pattern occurs.

Examples: **1-8\d\d-GOOG-411** will be expanded to strings:

1-800-GOOG-411, **1-801-GOOG-411**, ... **1-899-GOOG-411**.

http://www.\n*.com will be expanded to strings like:

http://www.a.com http://www.a123.com ... **http://www.ABCDefgHIJKLMNop.com**

Note: In choosing which patterns to include please be aware of the fact providing very generic patterns will make tesseract run slower.

For example, **\n*** at the beginning of the pattern will make Tesseract consider all the combinations of proposed character choices for each of the segmentations, which will be unacceptably slow.

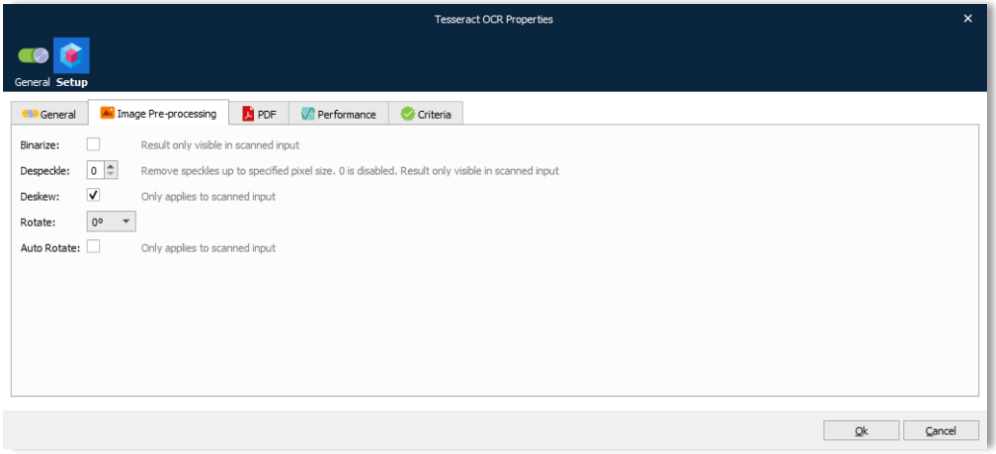
Because of potential problems with speed that could be difficult to identify, each user pattern has to have at least kSaneNumConcreteChars concrete characters from the unicharset at the beginning.

Languages One or more languages must be selected to specify the language(s) of the incoming data.

The number of loaded languages is limited only by memory, with the caveat that loading additional languages will impact both speed and accuracy, as there is more work to do to decide on the applicable language, and there is more chance of hallucinating incorrect words.

10.34.2 Image Pre-processing

Pre-processing of images is useful for scanned images, or images which Tesseract has trouble reading due to lack of contract between colors etc.

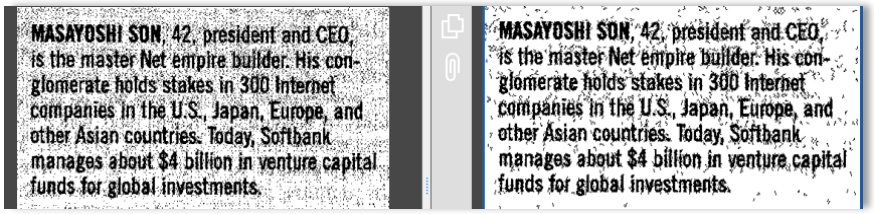


Binarize

Turns images monochrome. Result is used but only visible in output of scanned input.

Despeckle

Removes noise from images up to the specified size of grain in pixels. Despeckle requires images to binarized. Result is used but only visible in output of scanned input.



Deskew

A scanned image might be skewed causing issues with quality of OCR.

Deskew will straighten images that has been scanned or photographed skewed.

Deskew only works on scanned input and will be auto disabled if text or paths are found.

Skewed:

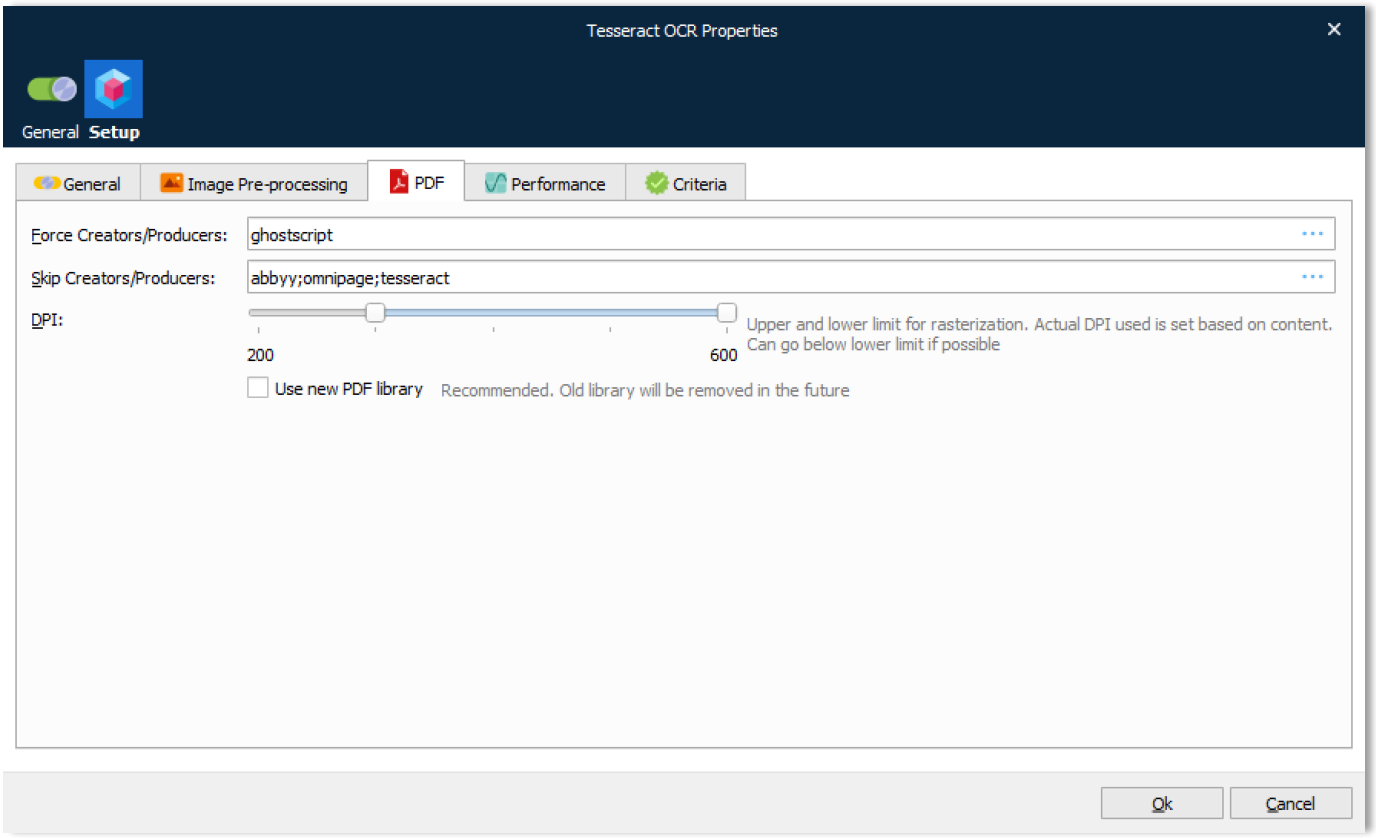
Deskewed:

Rotate	Rotate pages clockwise. Useful when input is always rotated the same. Will disable the Auto Rotate feature.
Auto Rotate	Scanned input will be automatically rotated based on the content of each page.

10.34.3 PDF specific

The Dots Per Inch (DPI) of an image input is given in the image file; however, a PDF can hold multiple images of different DPIs.

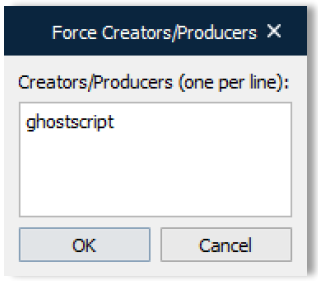
When PDF is used as input, the text layer is stripped, and the remaining images and paths (basic vector graphics) are rasterized. OCR is performed on the rasterized image.



Force Creators/Producers

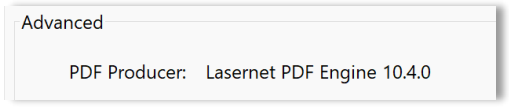
Tesseract can be forced to process an OCR rendering, for the full PDF document, which is required for producers like GhostScript (default value).

You can enter one or several producers by clicking the **...** button and entering one producer per line:



Notes:

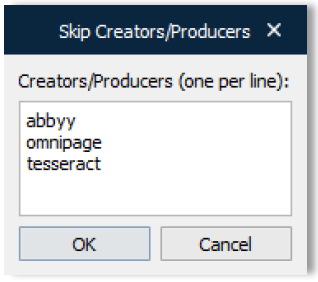
- The name is not case-sensitive
- The PDF Producer is detected in the Advanced settings of the PDF document.



Skip Creators/Producers

Tesseract can be skipped to process OCR rendering for the full PDF document. This is recommended if the PDF document is already processed by another OCR application, such as ABBYY, OmniPage or Tesseract (default values).

You can enter one or several producers by clicking the **...** button and entering one producer per line:



Notes:

- The name is not case-sensitive.
- The PDF Producer is detected in the Advanced settings of the PDF document.

DPI

The range of DPIs allowed in images used for OCR.

If the default DPI range is between 300 and 600 (default) it means that the DPI will never go above 600 but it might go below 300 depending on the existence of paths.

Note: Vector graphics have no resolution and no DPI.

Examples:

A scanned PDF containing pages with images of 1200 DPI.
This is above 600, which defines the upper limit of DPI range, and the image will be rasterized at 600 dpi. This upper limit has been chosen because higher values do not give a better result, but do take longer to process.

A standard PDF containing two images where first one is 150 DPI and second is 400 DPI.
To retain the image quality of the 400 DPI, the image for the OCR process will be rasterized at 400 DPI. This does not improve readability of the 150 DPI image, but it does take longer to process – that is the trade-off.

A standard PDF containing a 150 DPI image and no paths (vector graphics).
The OCR image is rasterized at 150 DPI because rasterizing at the lower limit of 300 will not improve readability but will slow down the process.

A standard PDF containing a 150 DPI image and paths (vector graphics).
Since paths exist, the OCR image will be rasterized at the lower limit of the defined range at 300 DPI. This is to ensure that any text written using paths will be readable.

Use new PDF library

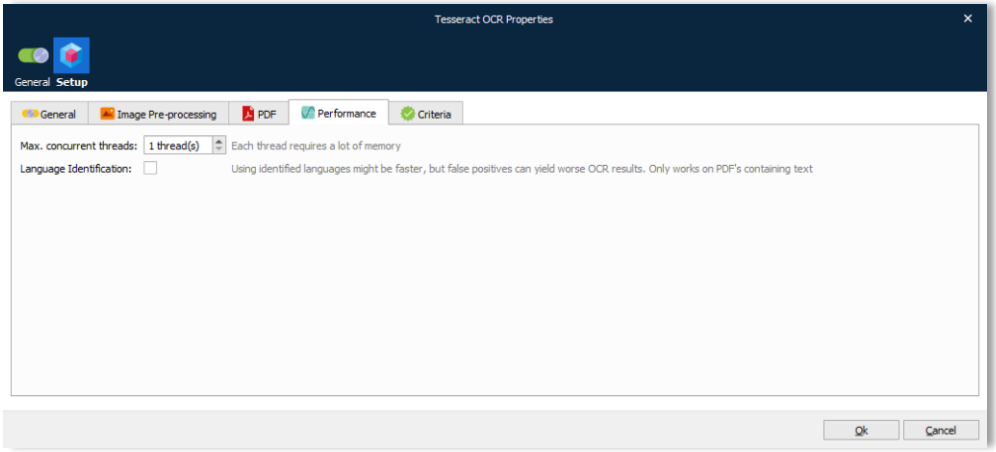
If selected, a library and algorithm will be used to improve the readability of text and images embedded in a PDF document. Several PDF producers do follow the PDF standard, and documents are detected as not valid by online validators but is still readable by most PDF readers like Acrobat Reader, Foxit Reader and DevExpress.

By introducing a new PDF library, the Tesseract OCR module can render PDF documents, that do not fully follow the PDF standard.

Note: We recommend that you select this setting for any new user of this module. When activated by existing users, the text output may not be expected to be 100% compatible with existing OCR Forms added to an OCR Engine, but it is still expected that the quality of the extracted text strings will be improved for most PDF producers.

10.34.4 Performance

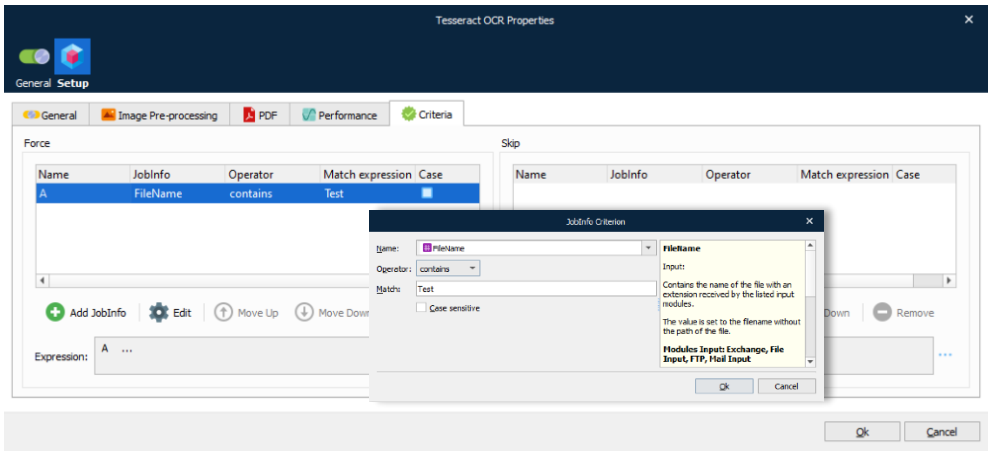
OCR is a CPU and memory intensive task. Per default Lasernet uses one thread per module instance to process all pages. This is to keep the memory usage as low as possible. However, it is possible to process more than one page in parallel per module instance, at the cost of higher memory usage.



Max. concurrent threads	Specifies the maximum number of concurrent threads tasked to perform OCR per Job per module instance. NB! An additional thread is spawned which creates the output, as OCR'ed pages become available.
Language Identification	The module will look for words in PDF and identify the languages used in the PDF, using .user-words files located in /TessData folder. Using only the languages identified, and not the full user selected list, will make OCR run faster and use less memory. NB! Language Identification is no better than the .user-words lists, and currently must be regarded as experimental

10.34.5 Criteria

In the Criteria tab you can add one or several criteria to manage the Force and Skip properties defined in the PDF tab. Add any JobInfo and an operator with a match to either force Tesseract to render the whole PDF document or skip to render any part of the PDF document.



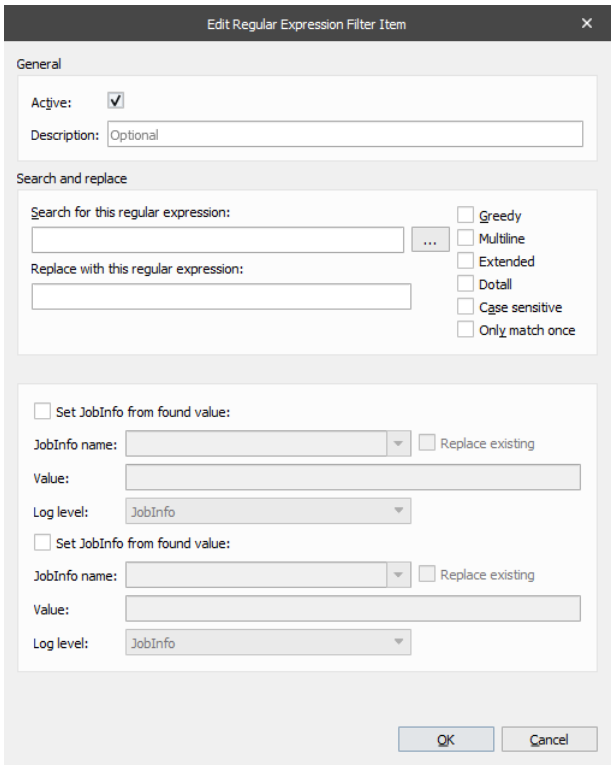
Force Producers	Tesseract can be forced to render the whole PDF document, for a defined list of producers, when a producer is detected in the advanced settings of a PDF Document.
Skip Producers	Tesseract can be skipped to render any part of the PDF document, for a defined list of producers, when a producer is detected in the advanced settings of a PDF Document.

10.34.6 Good to know

- Recognizing of handwriting or barcodes are not supported.
- If a document contains languages not specified in configuration, results may be poor.
- It is not always good at analyzing the natural reading order of documents. For example, it may fail to recognize that a document contains two columns and may try to join text across columns.
- Poor quality scans may produce poor quality OCR.
- It does not expose information about what font family text belongs to.
- Bounding boxes created for text found by OCR, are not correctly sized causing issues with painting on PDF in Lasernet OCR Editor, if metadata is extracted by PDF to Text modifier.

10.35 Text Filter

This modifier duplicates the functionality of the Text Filter Engine. Please see section for ***Text Filter Engine*** for further information.



10.36 TIFF

TIFF module is used to convert EMF or PDF data to TIFF format and combine a mix of EMF and PDF into a single TIFF.

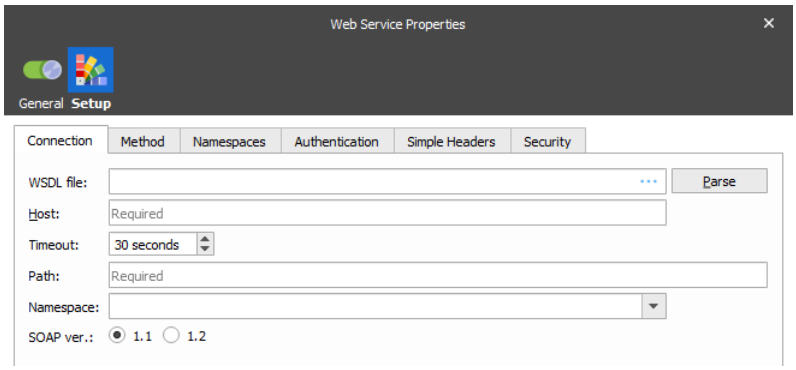
Paper size and orientation, in the incoming data, are automatically handled by the TIFF Modifier, by scanning the paper formats included in the EMF or PDF data.

10.36.1 Settings

See section for TIFF Engine for more information.

10.37 SOAP Web Service

Like its siblings, the Web Service modifier is used for retrieving or sending data from a Web Service. See Section *Input Modules → SOAP Web Service* for further information.



10.38 XML Signer

By signing your XML file using a digital certificate, you allow the recipient to verify both the origin and the integrity of the file. This means that the recipient is able to verify that the XML came from your organization and has not been changed since it was created.

10.38.1 Settings

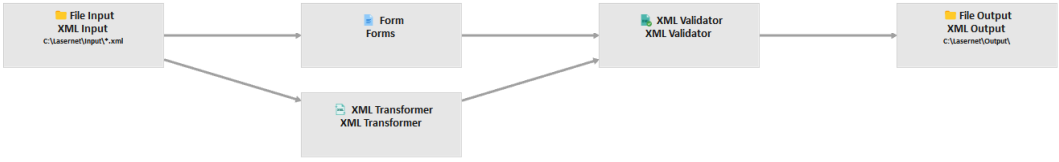
See section for XML Signer Engine for more information.

10.39 XML Validator

Validation of XML via XML Schema (XSD) is a way to verify that the data structure conforms to the associated schema before delivering it to another system.

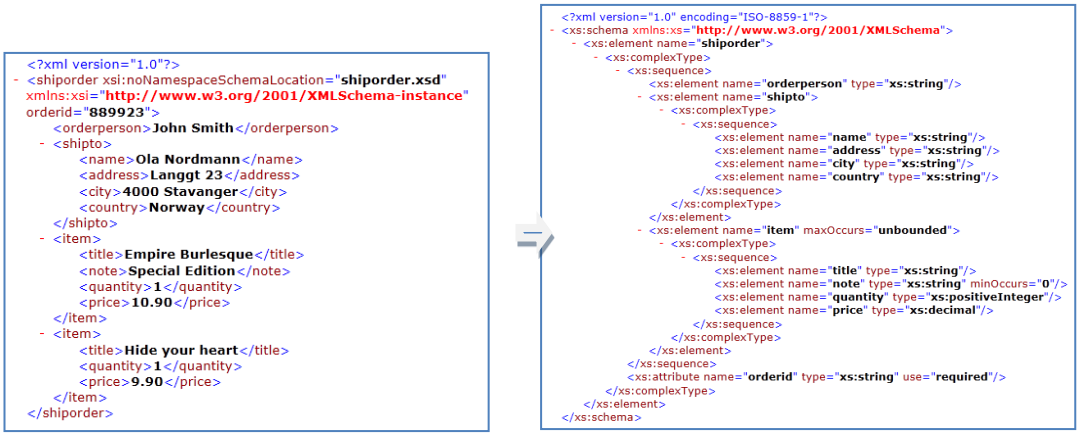
The XML Validator Engine or Modifier is able to parse and validate the XML data by using schema files stored locally or via a URL.

XML data can be received or created by any module and validated before parsing data to another module.



XML Schemas are primarily used to validate the structure and data types of the XML data. This covers:

- Which data elements or attributes are expected.
- What order the data elements are expected in.
- The nesting these data elements can have.
- Which data elements are optional and which data elements are required.



```
<?xml version="1.0"?>
- <shiporder xsi:noNamespaceSchemaLocation="shiporder.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  orderid="889923">
      <orderperson>John Smith</orderperson>
    - <shipto>
        <name>Ola Nordmann</name>
        <address>Langgt 23</address>
        <city>4000 Stavanger</city>
        <country>Norway</country>
      </shipto>
    - <item>
        <title>Empire Burlesque</title>
        <note>Special Edition</note>
        <quantity>1</quantity>
        <price>10.90</price>
      </item>
    - <item>
        <title>Hide your heart</title>
        <quantity>1</quantity>
        <price>9.90</price>
      </item>
  </shiporder>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    - <xs:element name="shiporder">
        - <xs:complexType>
            - <xs:sequence>
                <xs:element name="orderperson" type="xs:string"/>
              - <xs:element name="shipto">
                  - <xs:complexType>
                      - <xs:sequence>
                          <xs:element name="name" type="xs:string"/>
                          <xs:element name="address" type="xs:string"/>
                          <xs:element name="city" type="xs:string"/>
                          <xs:element name="country" type="xs:string"/>
                        </xs:sequence>
                      </xs:complexType>
                  </xs:element>
              - <xs:element name="item" maxOccurs="unbounded">
                  - <xs:complexType>
                      - <xs:sequence>
                          <xs:element name="title" type="xs:string"/>
                          <xs:element name="note" type="xs:string" minOccurs="0"/>
                          <xs:element name="quantity" type="xs:positiveInteger"/>
                          <xs:element name="price" type="xs:decimal"/>
                        </xs:sequence>
                      </xs:complexType>
                  </xs:element>
              </xs:sequence>
              <xs:attribute name="orderid" type="xs:string" use="required"/>
          </xs:complexType>
      </xs:element>
  </xs:schema>
```

XML Data sample

XML Schema sample

10.39.1 Location

Provide the XML schema in a separate XSD file.

The XML to be validated must contain an `xmlns:xsi` attribute and either an `xsi:noNamespaceSchemaLocation` or `xsi:schemaLocation` attribute (as appropriate) to specify the XSD to use.

The value of `xsi:noNamespaceSchemaLocation` or `xsi:schemaLocation` in the XML must be set as follows:

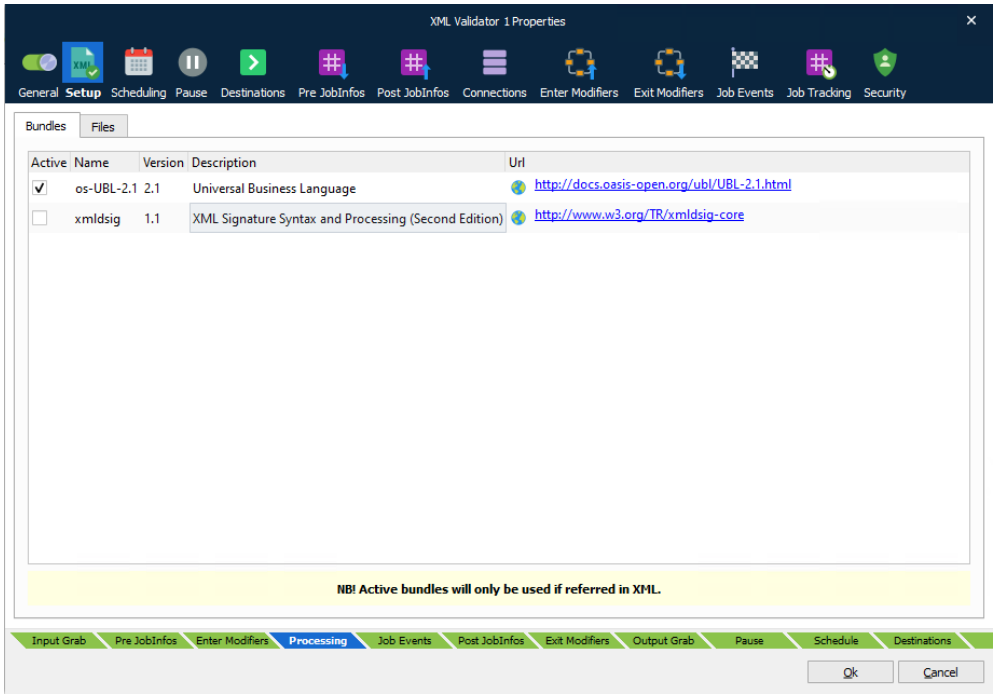
- **To identify a remotely accessible XSD:** A pair of full URIs. For example: `xsi:schemaLocation="http://rep.oio.dk/ubl/xml/schemas/0p71/pie/ http://rep.oio.dk/ubl/xml/schemas/0p71/pie/pieStrict.xsd"`

- **To identify an XSD schema file that has been added to the list of schemas specified in this XML Validator Engine or Modifier (as described below):** An XSD file name. For example, `xsi:noNamespaceSchemaLocation="shiporder.xsd"`. The XML Validator's **Files** tab contains a list of schemas that have been selected and added to this XML Validator. Lasernet copies these schemas to the **Schemas** folder of the Lasernet configuration's Resources area. Each item on the **Files** tab can have criteria that specify when that particular schema is used. The XML for validation must specify which of these schemas to validate the XML against.

10.39.1.1 Specify Schemas to Use from Supplied Bundles

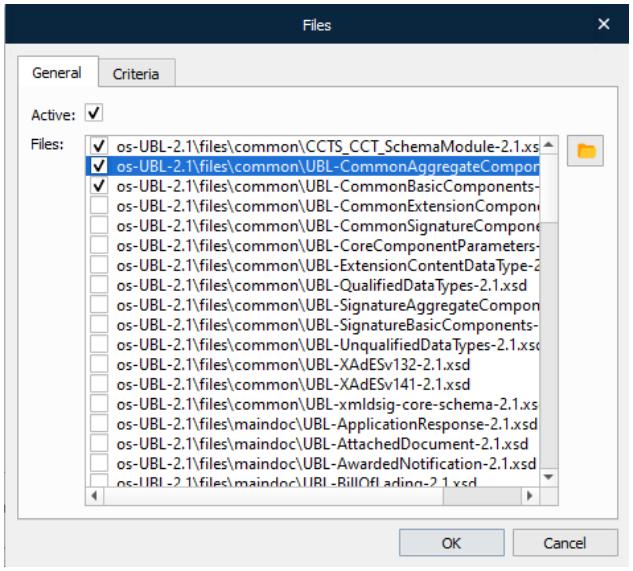
Click the **Bundles** tab if you want to use schema files that are bundled with Lasernet. In the **Active** column, select a bundle, then click **Ok**. The schema files in the selected bundle are imported in the **Schema** folder of the configuration's Resources area and afterwards are available for you to add to the configuration of this XML Validator.

Note: The **xmldsig** bundle enables this module to validate XML against `xmldsig-core-schema.xsd`. However, XML Validator will not validate the certificate within the XML.



Return to the XML Validator **Properties** window, go to the **Setup** tab, and then click the **Files** tab. Click **Add**, then in the **Files** window select checkboxes to specify one or more bundled schema files to add to this XML Validator's schema list (see image below). Use the **Criteria** tab to specify when these schemas are used.

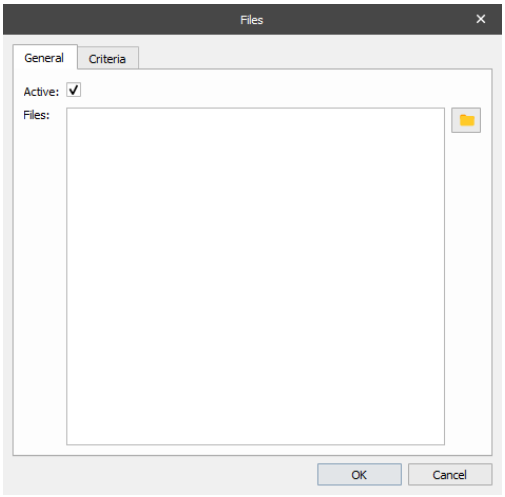
After you click **OK**, the files are added to the schema list on the XML Validator's **Files** tab. Click **OK** on the XML Validator **Properties** window to save your changes.

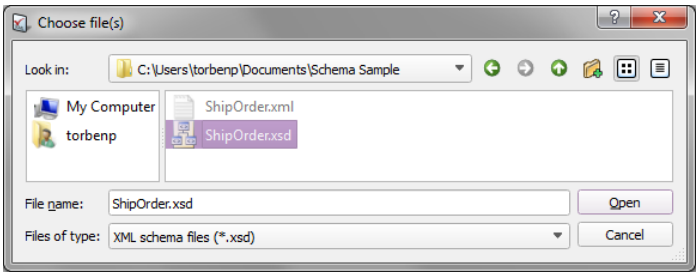


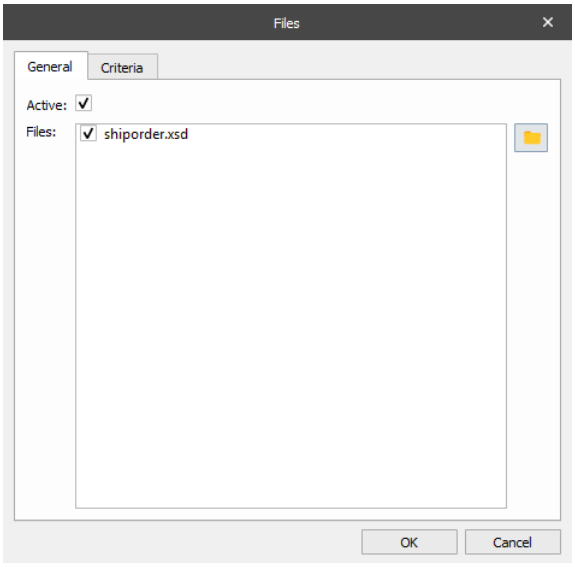
10.39.1.2 Supply Schemas Files to Use

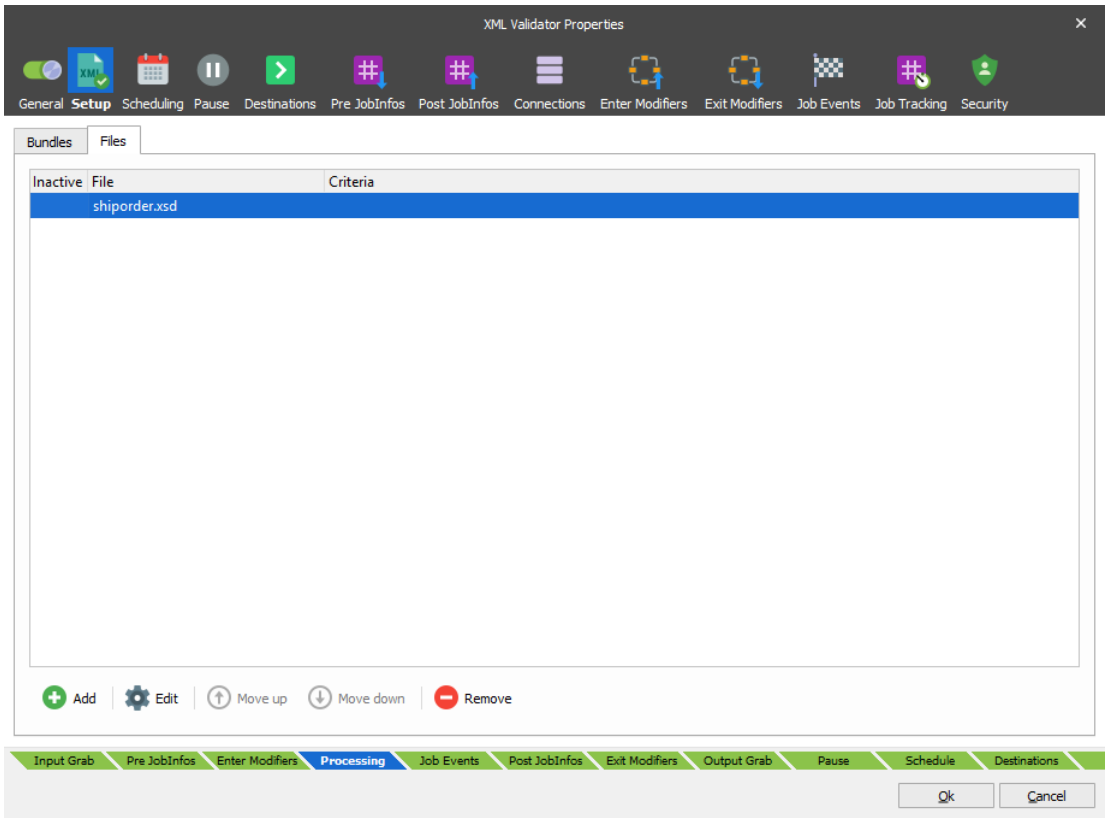
The **Browse** button in the **Files** window (beside the **Files** area) enables you to browse for XSD files outside the **Schemas** folder. The selected files are copied to the **Schemas** folder, so beware of creating duplicates.

After you have chosen files, Lasernet adds them to the **Files** area of the window. Select checkboxes in the **Files** area to specify which schema files to import into the configuration and add to the schema list for XML Validator. Use the **Criteria** tab to specify when these schemas are used. After you click **OK**, Lasernet imports the files.




Choose file(s)
Look in:
C:\Users\torbenp\Documents\Schema Sample
My Computer
torbenp
ShipOrder.xml
ShipOrder.xsd
File name:
ShipOrder.xsd
Open
Files of type:
XML schema files (*.xsd)
Cancel


Files
General
Criteria
Active:
Files:
shiporder.xsd
OK
Cancel


XML Validator Properties
General
Setup
Scheduling
Pause
Destinations
Pre JobInfos
Post JobInfos
Connections
Enter Modifiers
Exit Modifiers
Job Events
Job Tracking
Security
Bundles
Files
Inactive
File
Criteria
shiporder.xsd
Add
Edit
Move up
Move down
Remove
Input Grab
Pre JobInfos
Enter Modifiers
Processing
Job Events
Post JobInfos
Exit Modifiers
Output Grab
Pause
Schedule
Destinations
Ok
Cancel

The list of accessible schemas files is loaded from the **Schemas** folder in the configuration.

xmlns:xsi + xsi:schemaLocation

When specifying a schema location, it is important to notice that Lasernet does not contain any URI resolving. This means that the schema must be accessible directly.

Non-working:

Example of a location which Lasernet cannot use;

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="urn:oasis:names:specification:ubl:schema:xsd:Invoice-2 UBL-Invoice-2.0.xsd"

Working:

Location which points to schemas in the real world:

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://rep.oio.dk/ubl/xml/schemas/0p71/pie/
http://rep.oio.dk/ubl/xml/schemas/0p71/pie/pieStrict.xsd"

The difference between the non-working and the working example is that additional processing is required to look up 'urn:oasis:names:specification:ubl:schema:xsd:Invoice-2' in order to find the actual location of the schema. This also applies to imports using XML Schema as well. Lasernet does not maintain or support this additional processing so it is up to the Lasernet administrator to make sure that the XML and XML Schemas are specified using the full paths.

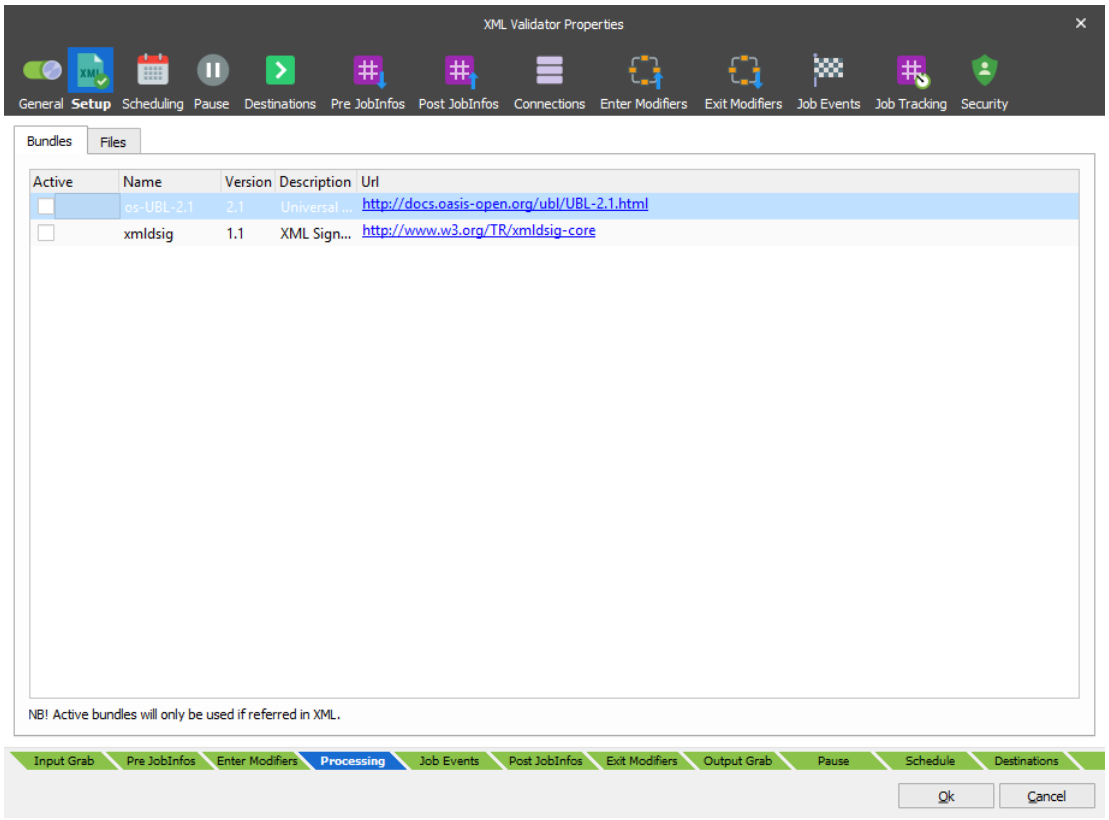
XML received from an external system should also be checked to ensure full paths are used.

Schemas folder

If no full path or URL is specified in the module, the schema is loaded from the **Schemas** folder in the configuration. It is important to note that schema which reference other files must do so using absolute paths, or relative to the Schemas folder.

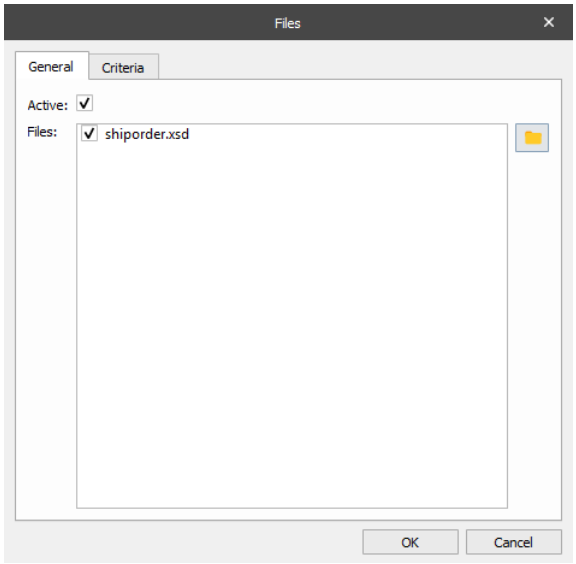
Engine

The XML Validator Engine fails the Job if validation is not successful. It also sets JobInfos so action can be taken accordingly.



10.39.2 Modifier

The XML Validator Modifier is not able to fail jobs. The JobInfo ‘XMLValidationError’ must be inspected after a call to the modifier.



JobInfos

The following JobInfos are set upon an unsuccessful validation either in the Engine or Modifier:

XMLValidationError	'1' if error has occurred
XMLValidationErrorMessage	Description of the occurred error

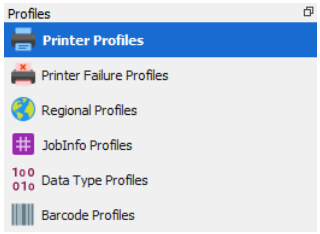
10.39.3 Running XML Validator as a Sheet Modifier

When running a XML Validator as a Sheet Modifier, typically at Sheet End in a form, the validation is only valid during the processing of forms in the Form Engine. Because of architectural limitations, it is not possible to execute and validate XML data in the output view of the Form Editor. When pressing Shift-F5 for evaluating modifiers in the Form Editor, the parsing will be based on input data only, which differs from the Form Engine which will correctly parse the XML data based on output data.

11 Profiles.

11.1 Printer Profiles

Note: If you are not going to use output printers, you can skip this chapter.



Printer Profiles are used to configure and setup printer settings. The Printer Profiles system has been carefully designed to interact with any printer driver that works with the Microsoft Windows printing subsystem.

Internally, Windows operates with a device setup description structure called the Device Mode structure, or DEVMODE for short. The DEVMODE describes everything that can be configured on a printer such as the paper format, orientation, duplex mode, substitution of true type fonts with device fonts, quality mode, etc.

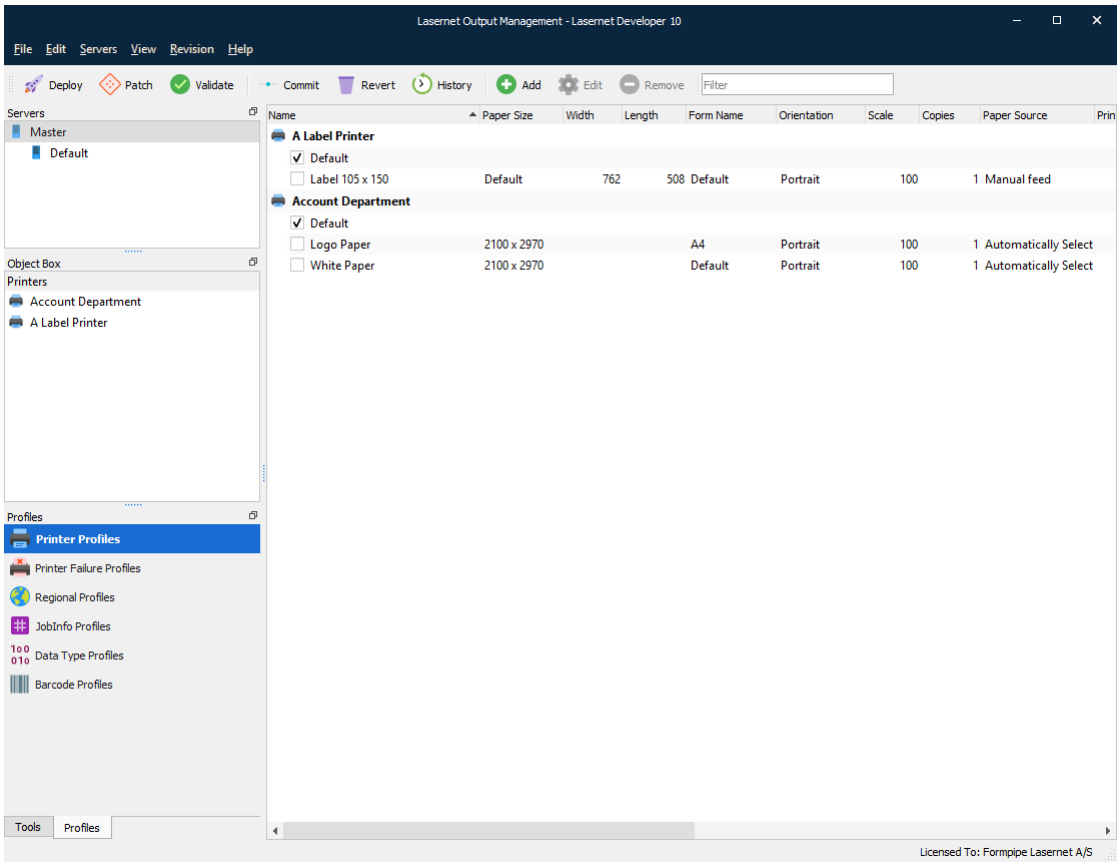
The DEVMODE structure is split into two distinctive parts. Certain settings are very common amongst printers and as such Microsoft created a standardized method of configuring those settings. Other settings are device specific and require a more complex setup. From a user perspective, the DEVMODE structure cannot simply be built using JobInfos. It has to be setup using the dialogs displayed by the printer driver. DEVMODE structures are also specific to each individual printer and cannot be shared between different printers.

To address these issues Lasernet uses the Printer Profiles system. A Printer Profile is a specific setup for a printer. Each output printer in Lasernet can have as many profiles created for it as needed. However, there is one special profile called “Default”. The default printer profile is the configuration setup in the Printers control panel in Microsoft Windows.

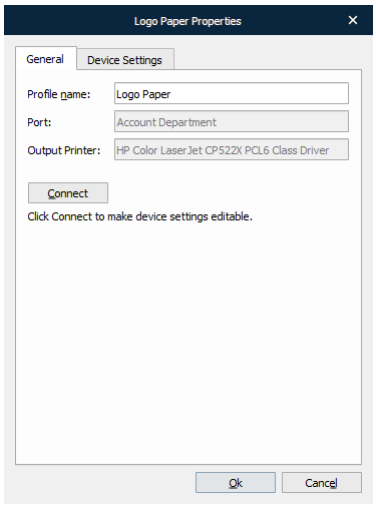
Printer Profiles are managed in the Lasernet Developer → Printer Profiles section.

Note: Printer Profiles are not objects and, therefore, cannot be seen in the object list when using "File - Export Objects". However, Printer Profiles are linked to the Printer Output objects for which they were created. When you export a Printer Output object, any associated Printer Profiles will be included with that object, making them available when you import the printer into another configuration.

You cannot select individual Printer Profiles under the printer object when exporting. All associated Printer Profiles will be included in the **.lnobjectx** export file.



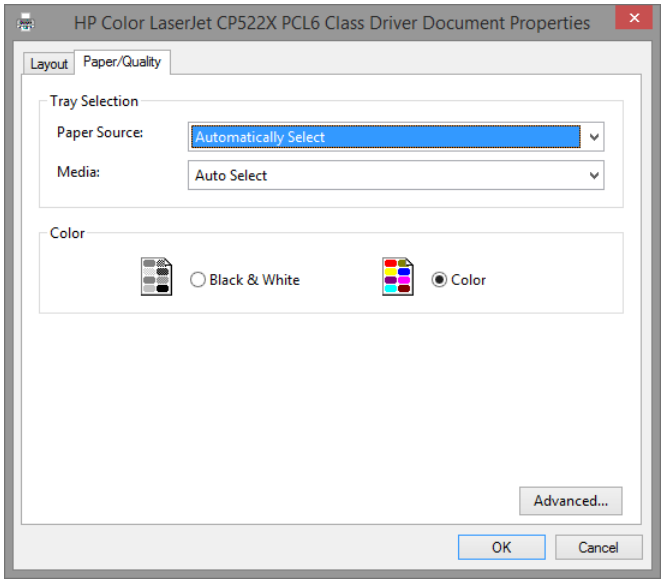
To create a new profile, click the **Add** button. Then click **Edit** on a profile to edit it. A dialog showing the settings that are common amongst all printers will be displayed:



Editing a printer profile requires Lasernet Developer to be able to connect to a specific printer. Due to the design of the Windows DEVMODE structure the printer driver has to be an exact match. If the versions do not match, all settings in the printer profile will be lost. To avoid accidentally altering a profile, first click the **Connect** button.

All the settings directly viewable in the dialog are part of the DEVMODE structure and can be changed using JobInfos or scripting. Special care has to be taken into account handling the Paper Source: the displayed names are localized. Meaning they are written in the language set by the regional settings and the printer driver. When specifying the paper source in a JobInfo it must match the language given in the regional settings/printer driver. This can become a problem in environments where mixed languages are used. This limitation is only specific to JobInfos, not if a paper source is selected in the dialog.

By clicking the **Additional Settings** button, printer driver-specific settings can be altered. This dialog will look different depending on what driver is selected. Below is an example of the settings for a Konica Minolta C360 Series PCL driver:



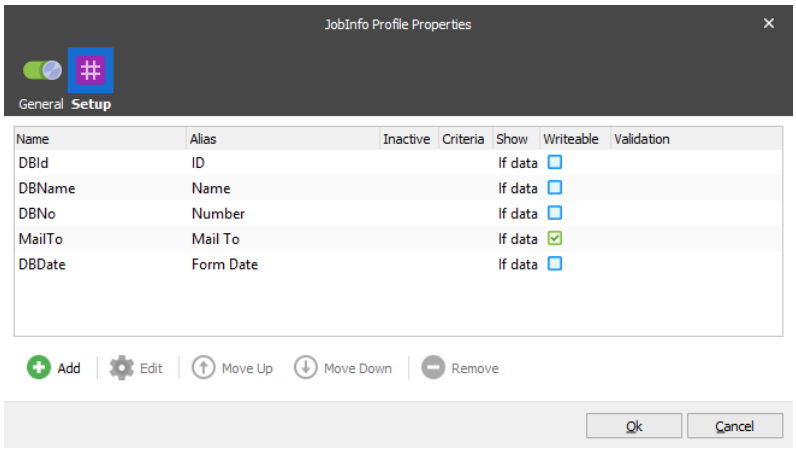
When a job is being printed in Lasernet, one printer profile is selected for the print. If the job itself does not select a printer profile, the selected radio button in the Printer Profiles page specifies it.

A profile can be selected for each printer. If a form is to be printed on this printer, then the selected profile will be used. For each page, different profiles can be used.

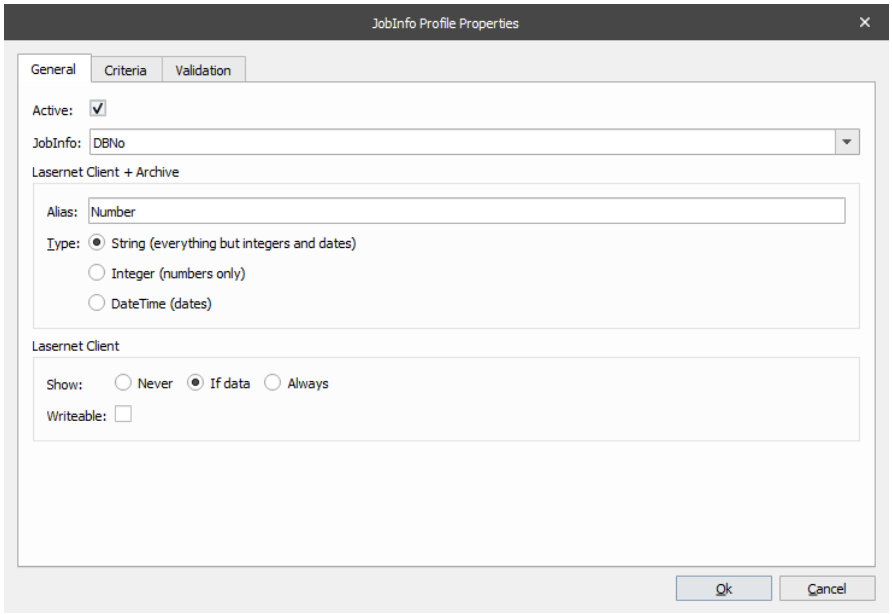
For instance, it is possible to take all pages except the last one from the Upper Paper Tray then take the last page from the Lower Paper Tray, assuming the printer supports this configuration. Another printer may allow paper to be taken from the Lower Paper Tray, use duplex mode and use Draft quality printing.

11.2 JobInfo Profiles

JobInfo profiles are used to display a list of relevant JobInfos in the Lasernet Client. Read the “Lasernet 10 – Client” manual for more information. Once configured, the key data is shown in columns for the job. JobInfo profiles can be used to show useful information such as invoice numbers or names and email addresses of clients, etc.



JobInfo names are not always user friendly, therefore it is possible to specify an alias if desired.

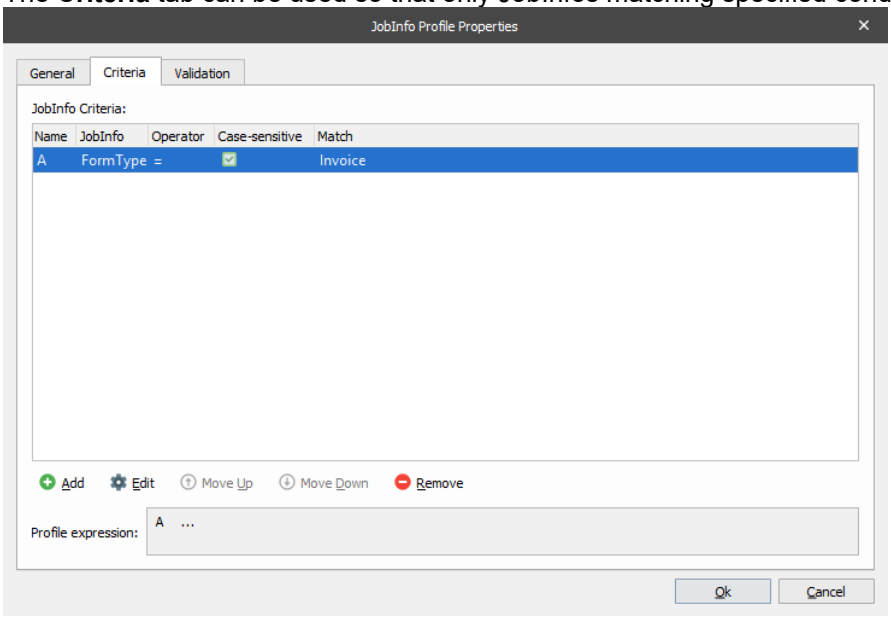


Three types of JobInfos are supported and handled accordingly: **String**, **Integer** and **DateTime**.

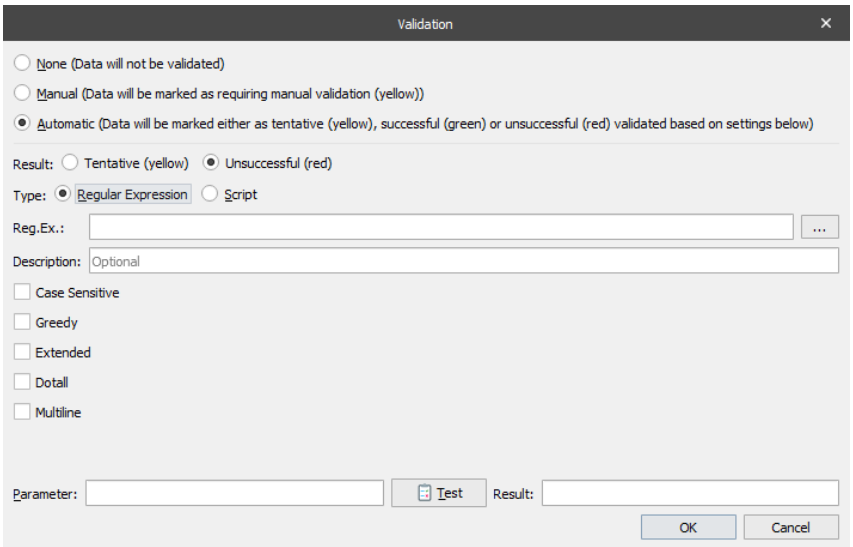
The **Show** property specifies whether or not a JobInfo should be shown as a column in the Client. As all stored JobInfos are searchable, it is not always necessary or desirable to make them visible in the Lasernet Client.

The **Writeable** property specifies whether or not a JobInfo is allowed to be edited in the Lasernet Client. As all stored JobInfos are searchable, it is not always necessary or desirable to make them editable in the Lasernet Client.

The **Criteria** tab can be used so that only JobInfos matching specified conditions are stored.

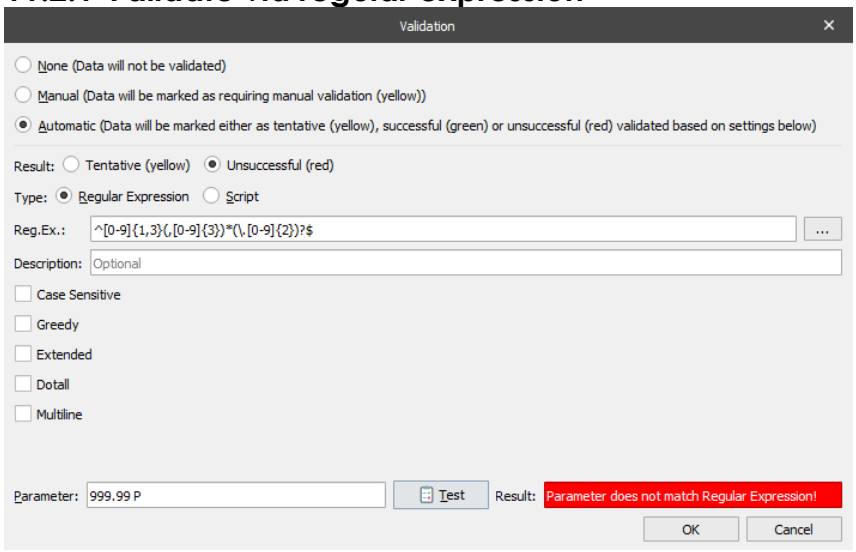


The **Validation** tab is used to validate the contents of a JobInfo in the Lasernet Client. JobInfos not matching specified conditions will be marked as either tentative (yellow) or unsuccessful (red).



Type a regular expression or a script to validate the contents of a JobInfo.

11.2.1 Validate via regular expression

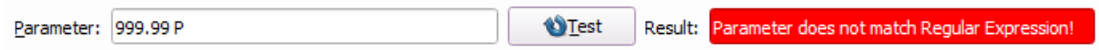


In this example we have created a regular expression matching a number in the US format which uses a comma as a thousand delimiter and a dot as comma delimiter.

Regular Expression: ^[0-9]{1,3}(,[0-9]{3})*(\.[0-9]{2})?$

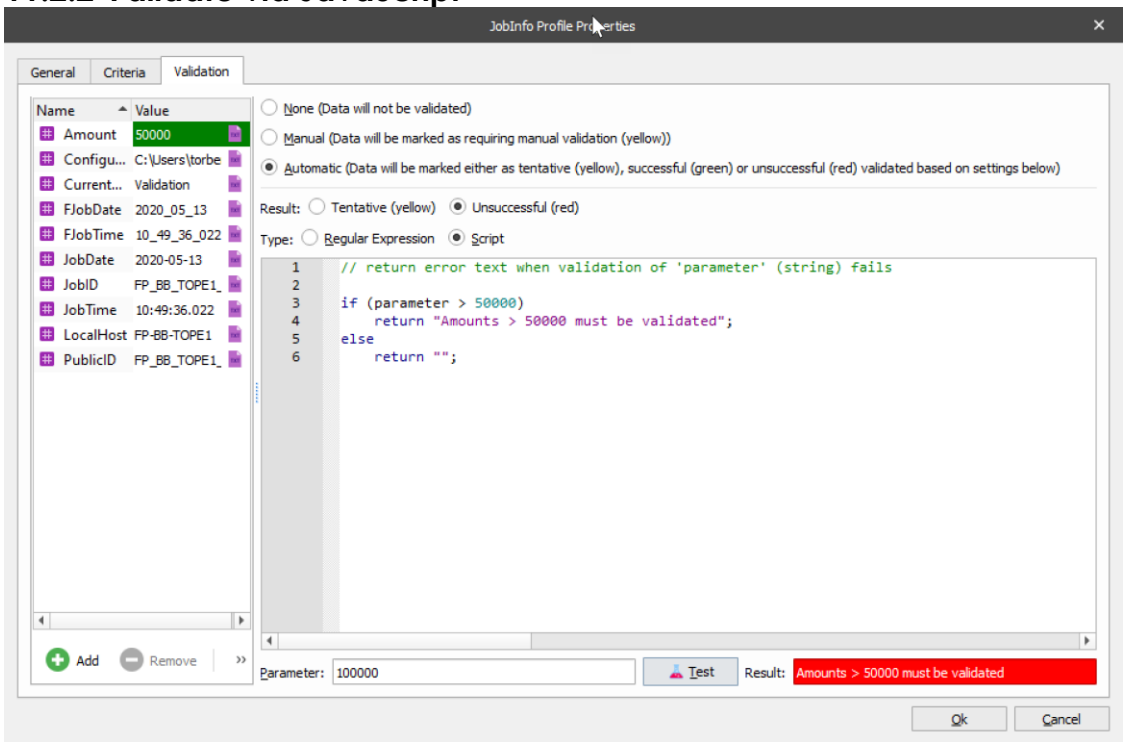
The $ sign at the end of the regular expression is essential to match the full string and not a part of the string.

The **Parameter field** and **Test button** are useful to validate the expected result of the regular expression. This functionality is for online testing only in the Lasernet Developer.



For more information regarding the syntax for regular expressions, we recommend searching the Internet.

11.2.2 Validate via JavaScript



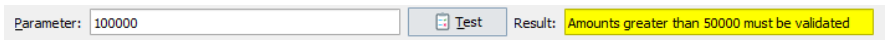
In this example we have created a JavaScript matching if a parameter is higher than a specific amount. The **parameter string** is essential and contains the contents of the JobInfo to be validated. The expression will return an empty string when validation of the parameter is successful and error text when it fails validation.

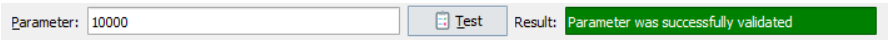
```
if (parameter > 50000)

return "Amounts > 50.000 must be validated";

else

                    return "";
```

If you want to validate against a **JobInfo** included in the script you can press **Add** to add the JobInfo **Name** and **Value** in the JobInfo grab window.

```
if (parseFloat(parameter) > parseFloat(job.getJobInfo("Amount")))

return "Amounts > 50.000 must be validated";

else

                    return "";
```

The **Parameter field** and **Test button** are useful to validate the expected result of the JavaScript. This functionality is for online testing only in the Lasernet Developer.





11.2.3 Supported JavaScript functions

The JavaScript tool in the validation dialog supports a limited range of system functions compared the Job Engine. The supported convert functions, job classes and arrays are:

Static functions:

- Convert.toDate
- Convert.toDateRP
- Convert.toNumber
- Convert.toNumberRP

Job class:

- All functions in the job class (see JavaScript manual section 4.2).

Arrays:

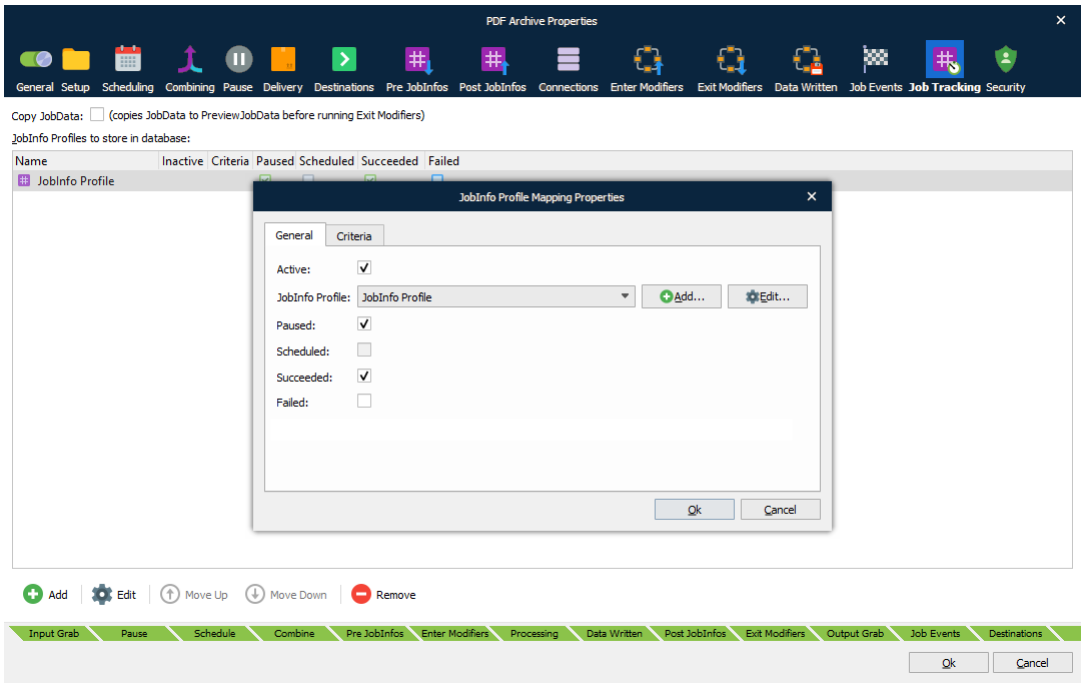
- Azure commands
- Database commands
- Modifier commands
- SharePoint commands

Note: The system functions are not supported for validation in Lasernet Meta, where validation runs on the client side instead of the server side.

11.2.4 Job Tracking

Add the list of JobInfo Profiles in the Module → Job Tracking tab. This will activate storage of metadata in the database for paused, scheduled, succeeded, archived or failed jobs.

When a module is configured to pause (Pause tab) or to use scheduling (Scheduling tab), it is possible to specify which JobInfos should be stored for later viewing in the Lasernet Client. This is achieved by first creating one or more JobInfo profiles and then mapping them to the module on the Job Tracking tab. More than one JobInfo profile can be added to the same queue and a JobInfo profile can be used on several different queues.

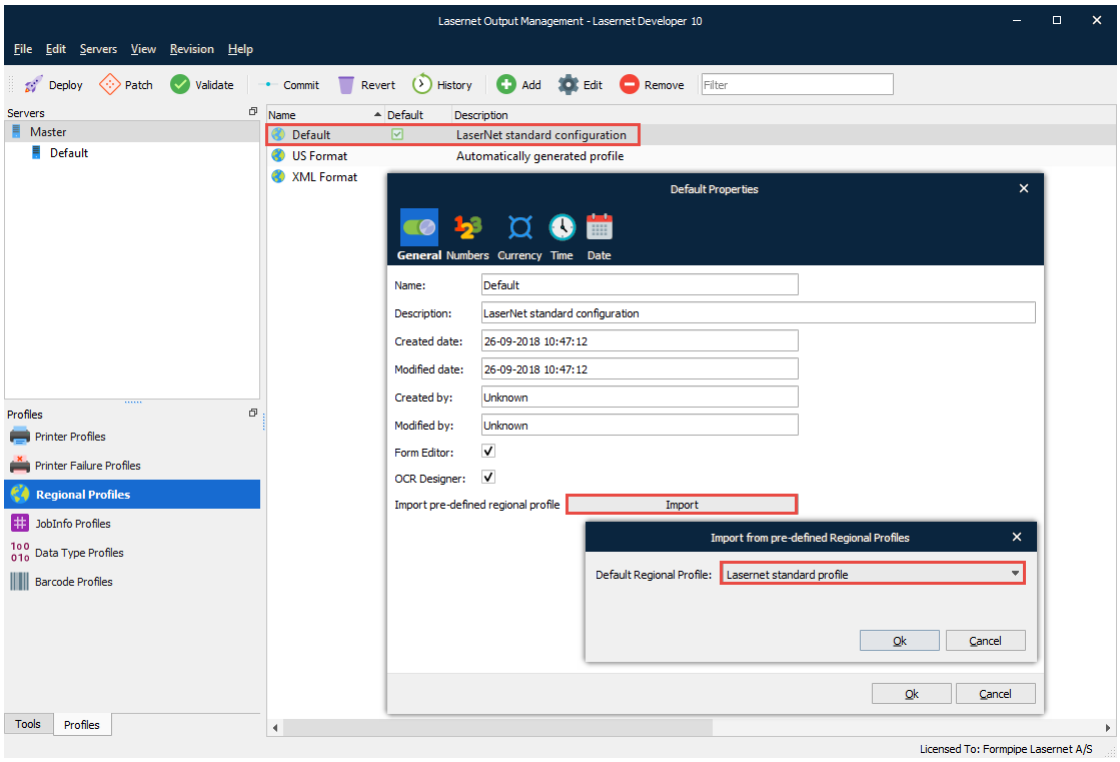


It is possible to use different JobInfo profiles based on criteria.

11.3 Regional Profiles

The regional profile settings look similar to Windows “Region and Language -> Formats” dialog. In a Regional Profile, you can specify number, currency, time and date formats to use in forms added to the Form Engine. You can also import a pre-defined Regional Profile.

In the **Regional Profiles**, you can select the **Default** regional profile as well as importing a pre-defined profile.

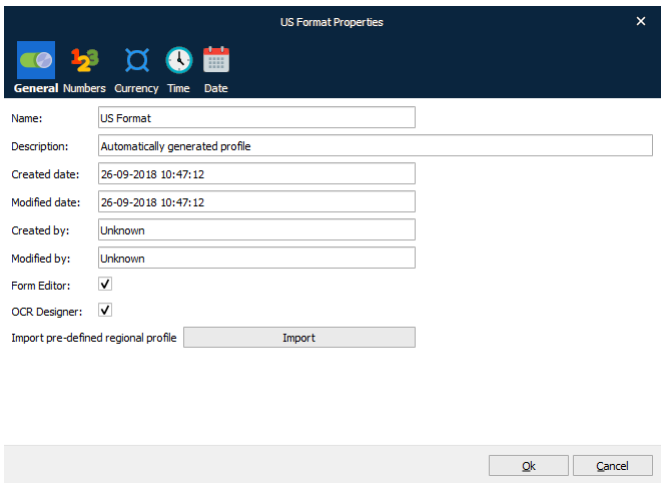


The default regional profile, named **Default**, is added to all new configurations. It can be changed, renamed and removed in the same way as other regional profiles that are manually added to the configuration.

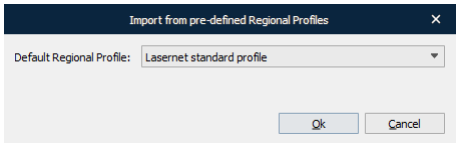
The default regional profile, marked with a ☑, will be used if no regional profile is assigned to a form, a sheet or a rearrange processed by the Lasernet Form Editor/Engine.

Information about how to use regional profiles is available from the Lasernet Form Editor Guide in the Formpipe Knowledge Base.

Click the **Add** button in the tool bar to add additional regional profiles to the configuration.



The **Import** button brings up a dialog with a dropdown list where the user can select pre-defined settings to be imported to the new profile.



After selecting a profile, the user can review and change the settings for **numbers**, **currency**, **time** and **date**. Pressing **Ok** will save the profile. Pressing **Cancel** or **Esc** closes the dialog without saving the profile.

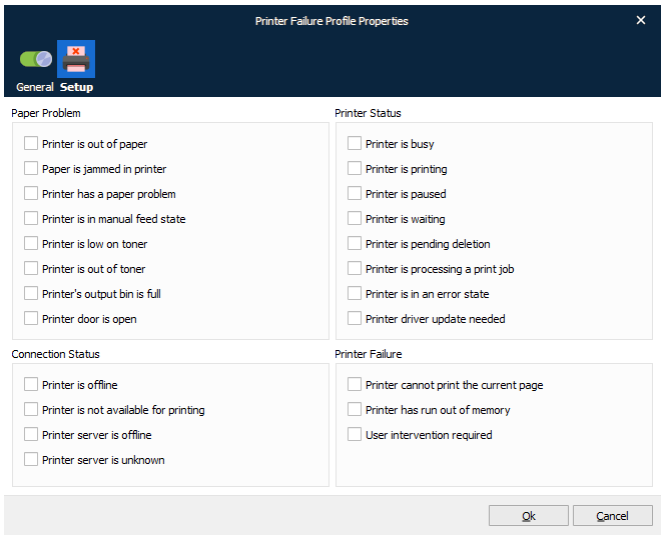
All configurations need at least one profile; therefore, it is not possible to remove the last remaining profile in a configuration.

Form Editor If activated this profile will be available in the Form Editor.

OCR Designer If activated this profile will be available in the OCR Designer.

11.4 Printer Failure Profiles

Lasernet is able to retrieve Windows printer queue status messages. These can then be used to help set up rules for redirecting a print file to another output module, if a network connection is down on a local printer or if toner is low, etc. This can help to minimise the number of failed jobs and provide redundancy when processing important print jobs.



By default, none of the settings are activated. Each status can be turned on/off.

The profile is stored with a user defined name and is accessible in the printer setup of the output printer (below the setting for the “External print timeout”).

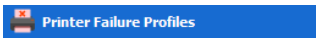
If one of the Printer Failure settings is true, the job will fail and the error log message will include the reason for the failure. The highest priority failure for a job is the output module not connecting to the Windows printer. All other failures are considered secondary.

11.4.1 Status on a printer queue

When sending print jobs to a Windows print queue via the Printer Output module, Lasernet is able to obtain the status from the printer queue. By default, Lasernet will only check if the spool queue is available and only fail the job if the Windows printer queue cannot be reached.

11.4.2 Setting up rules for failing jobs

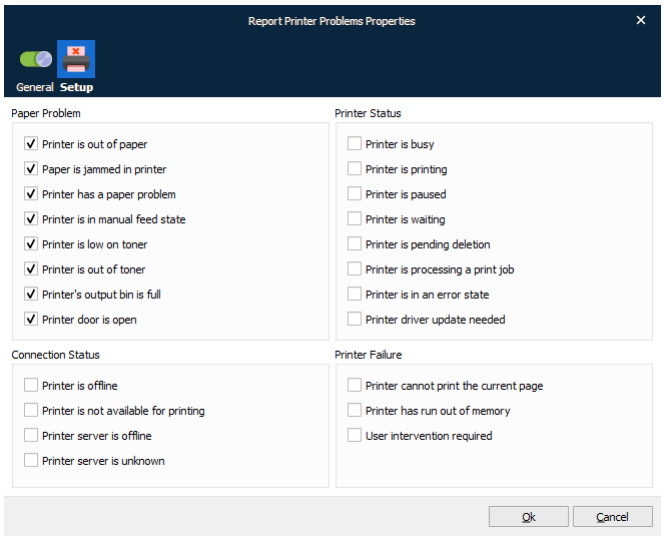
In the Printer Failure Profiles you can set up unique rules for how print jobs should fail depending on the status of specific printers.

Go to Printer Failure Profiles 

for adding, editing or deleting your profiles.

If you want to use different rules for failing jobs on individual printers, you can add several profiles containing the exact settings for each printer.

The image below shows an example printer failure profile that will only handle paper problems. In practice, a failure profile *could* be configured to handle any mix of the properties available.



Click on any number of status types to create your failure profile and then click **Ok**.

We recommend using names which describe exactly how the profile behaves when failing a job. Alternatively, you could use printer names to help keep track of which profile is used for which printer module.

11.4.3 Status types

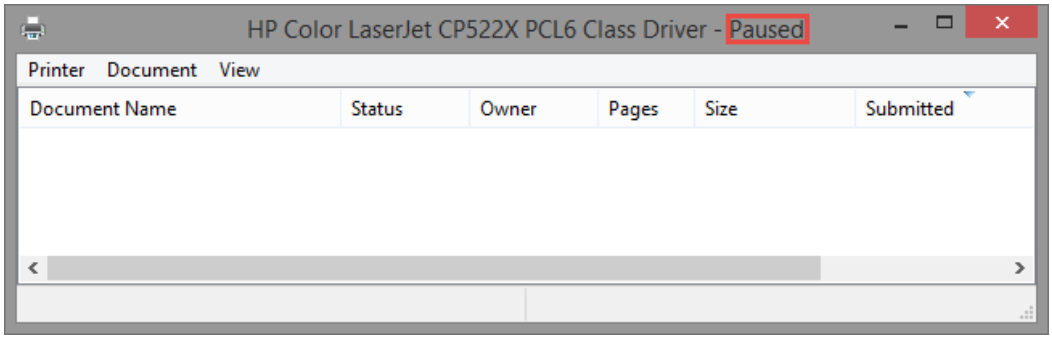
Status types are grouped in to four types of error:

Paper Problem	Issues related to handling of paper in the physical printer.
Printer Status	Issues related to processing of jobs.
Connection Status	Issues related to connectivity.
Printer Failure	Issues related to internal failures in the physical printer.

11.4.4 Caution

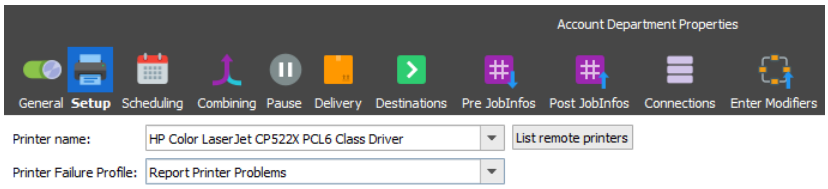
Lasernet is only able to obtain status messages from the Windows printer queue if the printer status has been reported to the Windows Spooler System.

Below is an example of 'Paused Mode' as reported to the printer queue.



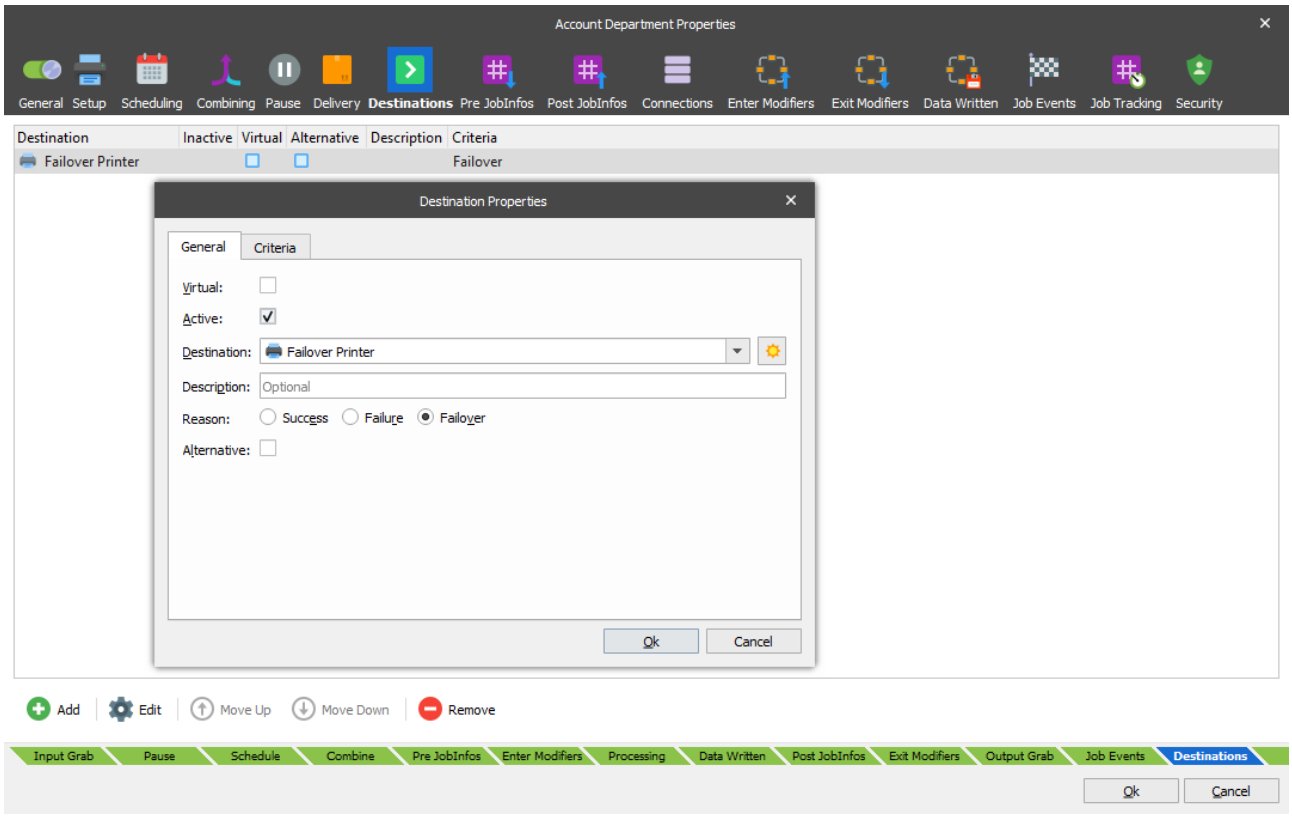
11.4.5 Adding Printer Failure Profile

In the Printer Setup tab for the Printer Output module, you can define which Printer Failure Profile you want to make use of.



If the status message is true for the given Printer Failure Profile during printing, Lasernet will fail the job and store the error message in the JobInfo called PrinterFailureMessage.

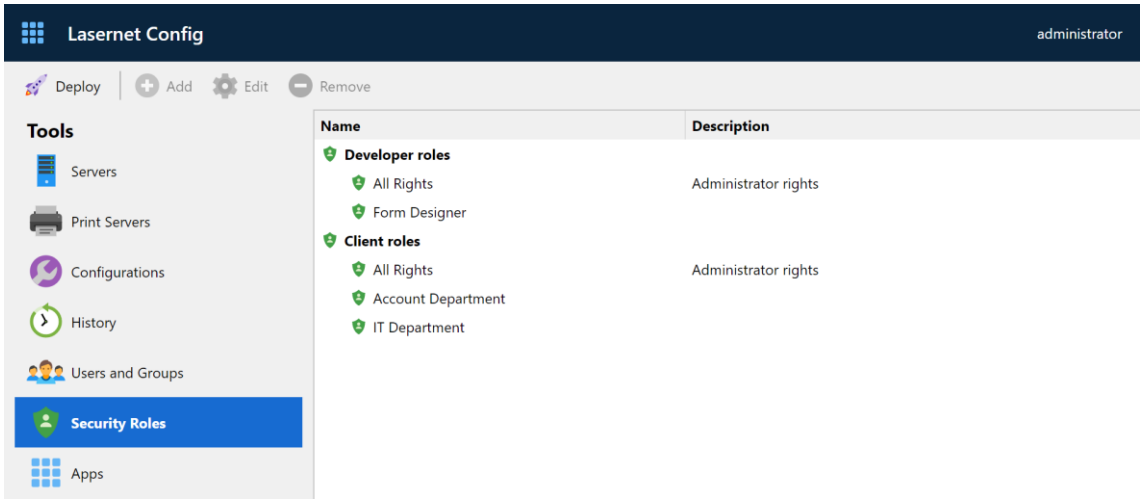
If the job fails and you want a print job to be redirected to another module, you can set up an alternative job destination on the printer module.



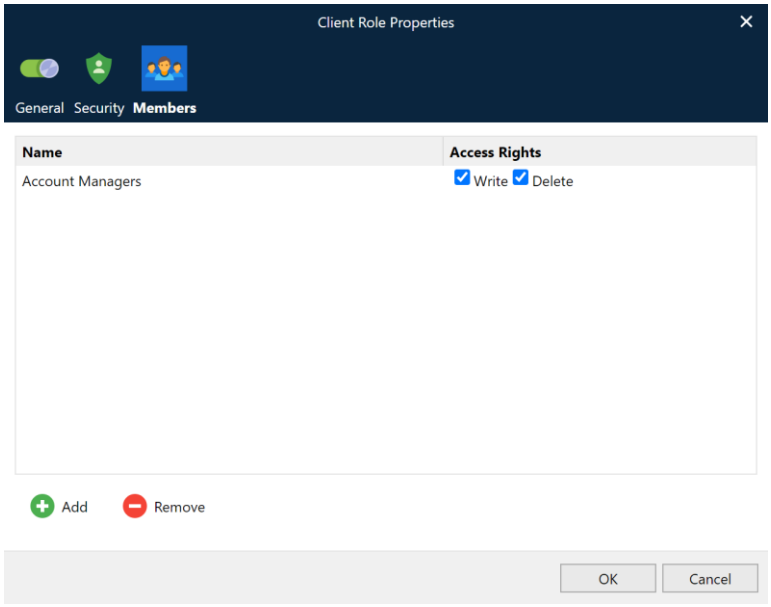
11.5 Security Roles

Security Roles are used for setting up authentication in Lasernet Client or Lasernet Developer and to set permissions on temporary saved documents. By default, they are visible for everyone in the Lasernet Client, but by adding Security Roles with specific read, write and delete permissions, it can help you to define which users or groups are allowed to view the individual documents.

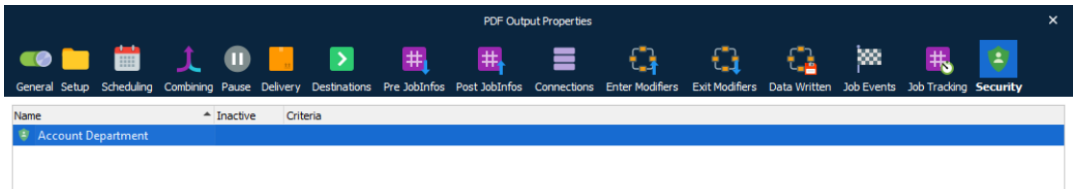
Go to the Lasernet Configuration Server and manage your profiles, groups and users.



Security Roles can include a user or a group and rights for document management in Lasernet Client.

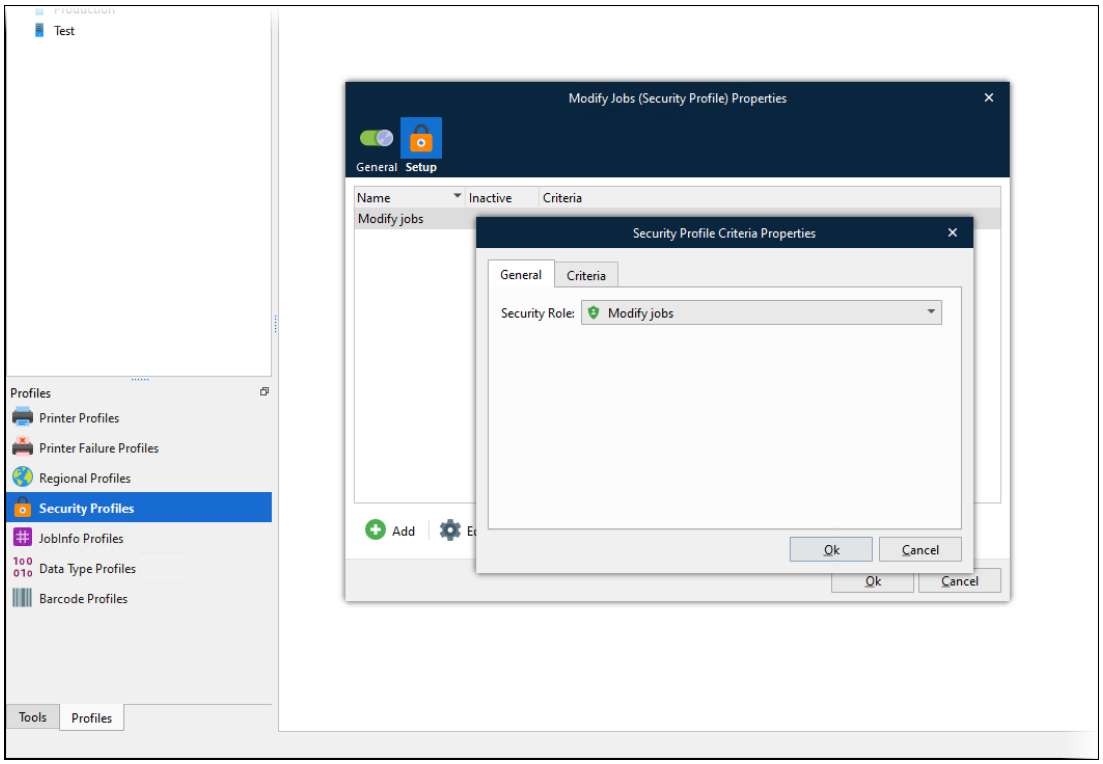


Security Roles can be added to any module available in the configuration. When the job is saved it will use Security Roles (if no Security Role is included, the job will be available to everyone). Roles are used to secure who is allowed to view and manage specific paused, scheduled or failed jobs in the Lasernet Client.

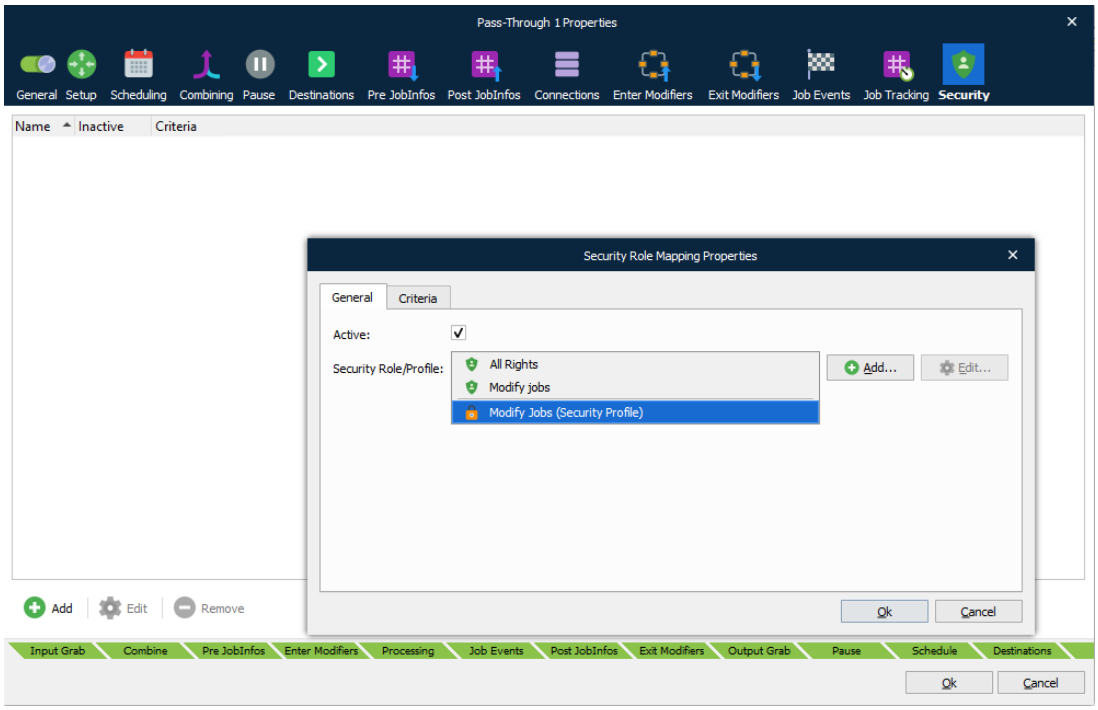


11.5.1 Security Profiles

Security Profiles enable you assign multiple security roles to a module as a single security profile. You can also add criteria to each security role assignment.



You can add security profiles to a module's **Security** tab (as well as security roles).



11.5.2 Users

Create a user login and a password and activate which application the user can login to. The user can belong to any group and Security Role in the configuration.

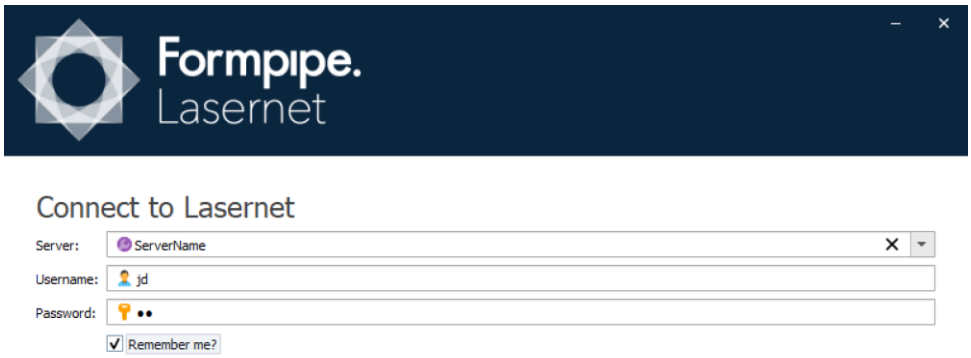
If removing a user from the configuration, any jobs saved by the user can only be managed by an administrator in the Client.

11.5.3 Groups

A group can contain as many users required and be removed if they do not belong to the group anymore. If removing a group from the configuration, any jobs saved by the group can only be managed by an administrator in the Lasernet Client.

11.5.4 Lasernet Developer

Any user added to the Security Roles in **Lasernet Config 10** server, with the Developer option activated in the user setup, is allowed to log into the Lasernet Developer and view and configure the setup. After adding a new user, the configuration needs to be re-opened in the Lasernet Developer. There must be at least one user added to Security Roles, with the Developer option activated, before you are prompted with a login window.



The user login for Lasernet Developer will enable you to Commit/Revert object. Any objects that are created or modified will have the user's name saved against them (see the General Tab for Created By/Modified By for objects).

12 Combiner and Scheduler.

12.1 Combining

Combining is available in the list of modules below and can be used to merge a list of jobs into a single job. The JobInfo named 'JobData' will contain the data for all the combined jobs. It will be attached to the JobInfos that were created in the first job. JobInfos for all other Jobs expect the first job will be deleted.

Some of the modules require additional properties depending on the input and output format.

Module	Type	Input Format	Output Format
Binary Merger	Engine	RAW	RAW
Compression	Engine	*	ZIP
EMF to RAW	Engine	EMF	RAW
Pass-Through	Engine	EMF, TEXT	EMF, TEXT
PDF	Engine	EMF	PDF
PDF Merger	Engine	PDF	PDF
TIFF	Engine	EMF	TIFF
XML Merger	Engine	XML	XML
Exchange	Output	**	**
File	Output	EMF, TEXT	EMF, TEXT
Mail	Output	**	**
Printer	Output	EMF	EMF

* **Compression Engine** can combine various job formats into a ZIP-archive for output. A filename for each input file is required and should be added to a user specified JobInfo.

** **Exchange** and **Mail** modules do not combine lists of jobs into a single job. Instead, a single email will be generated containing several attachments.

EMF format can be combined into a multi-page **EMF**, **PDF** or **TIFF** job.

TEXT format can be concatenated into a single **TEXT** job.

XML format can be merged into a single **XML** job, but requires special XPath properties.

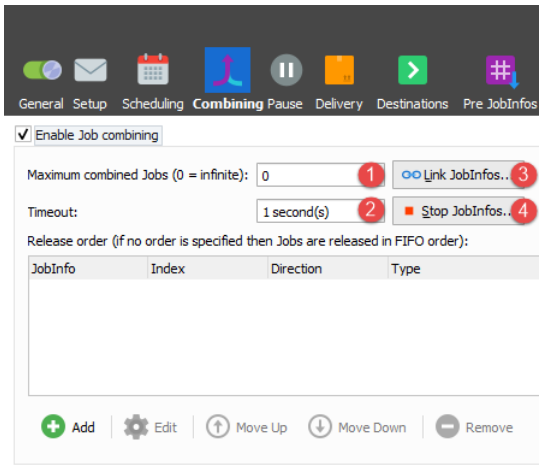
12.1.1 Setting up combining

Select the combining tab and activate **Enable Job combining**.

The default settings for combing are: **Maximum combined Jobs** equal zero, **Time out in seconds'** equal one and **Link JobInfos** and **Stop JobInfos** are left empty;

Using the default settings, Lasernet is only able to combine jobs if the input job is received as a single job. If Lasernet receives multiple jobs on same input module, a Link JobInfo or Stop JobInfo is required.

Lasernet is not able to combine jobs from a mix of input modules, all jobs have to be from same input module.



There are several ways to enable or empty combiner queues.

Maximum combined Jobs (1)

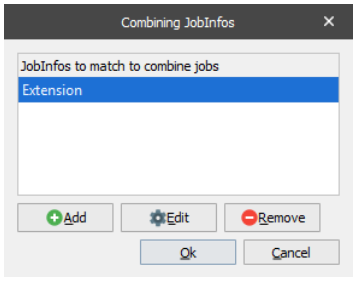
The queue will be emptied each time the number of jobs is in the queue reaches the figure set here (other than zero).

Time out in seconds (2)

If the given amount of seconds pass without any new jobs coming in then the queue is emptied.

Link JobInfos (3)

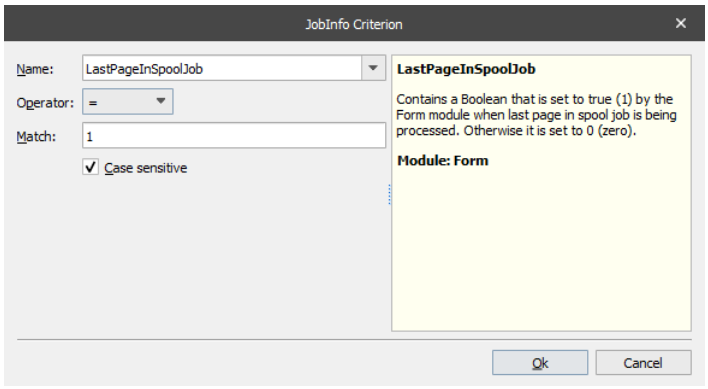
Selecting **Link JobInfos** criteria means that all the JobInfos have to be equal for them to end up in the same queue. For example, if you combine using the JobInfo Extension, only jobs with the same Extension will be combined.



The parameters for **Maximum combined Jobs, Time Out** and **Stop JobInfos** have a higher priority and will empty the queue even if **Link JobInfos** is true.

Stop JobInfos (4)

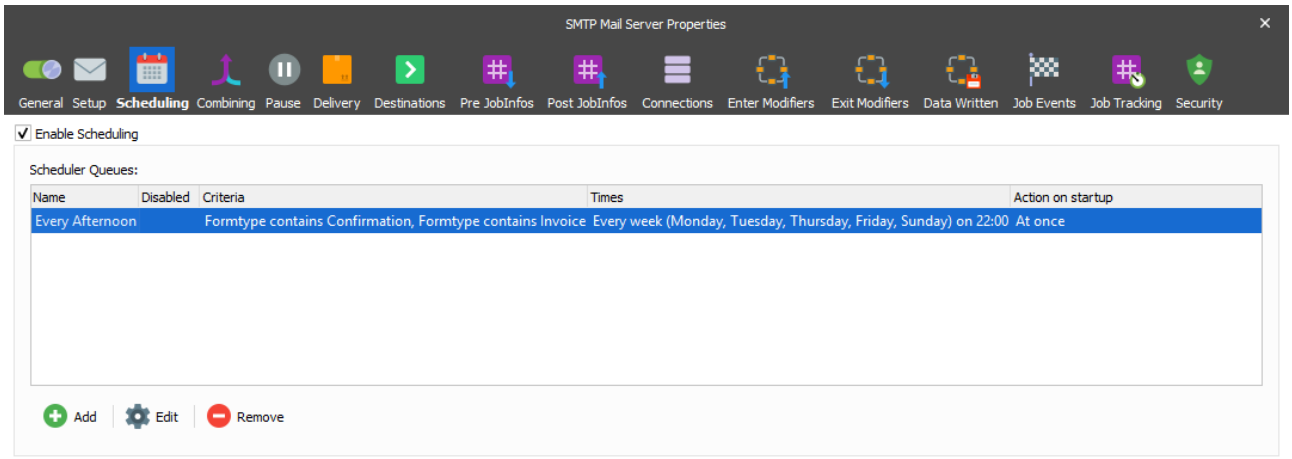
If a listed expression is true, the queue is emptied. For example, if you enable combining using an expression that LastPageInSpoolJob equals 1, any job will be combined until expression is true.



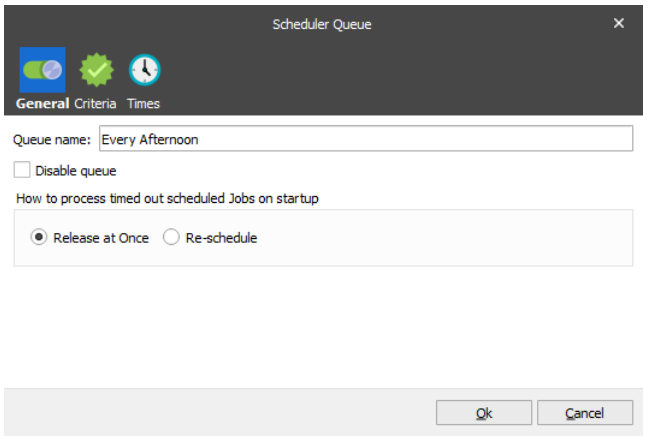
12.2 Scheduling Jobs

Rather than continuously sending jobs out of Lasernet, scheduling can be used to send them out at predefined times. When a module supports scheduling, its **Properties** window contains a **Scheduling** tab.

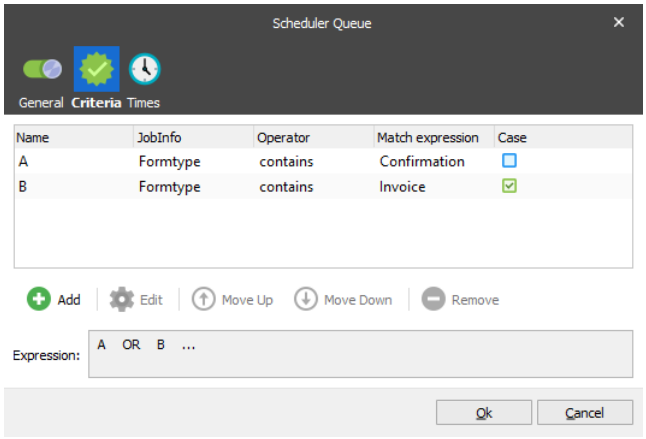
Note: All times are specified in UTC (Universal Time Coordinated).



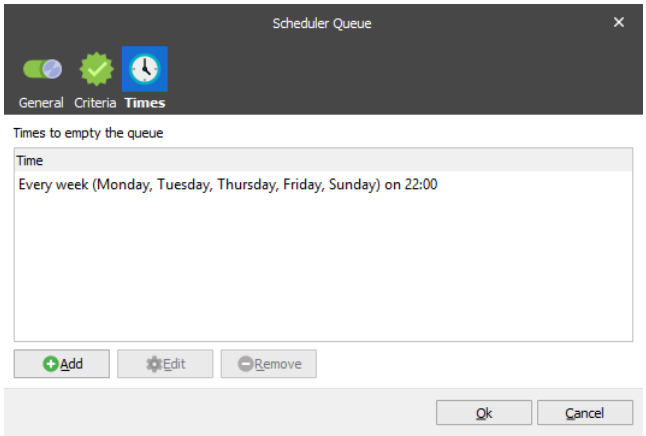
Double-click an entry on the **Scheduling** tab or click **Edit** or **Add** to configure a scheduler queue.



On the **General** tab, enter a unique **Queue name**. When a queue is emptied by Lasernet, a JobInfo SchedulerQueue is set using this name.



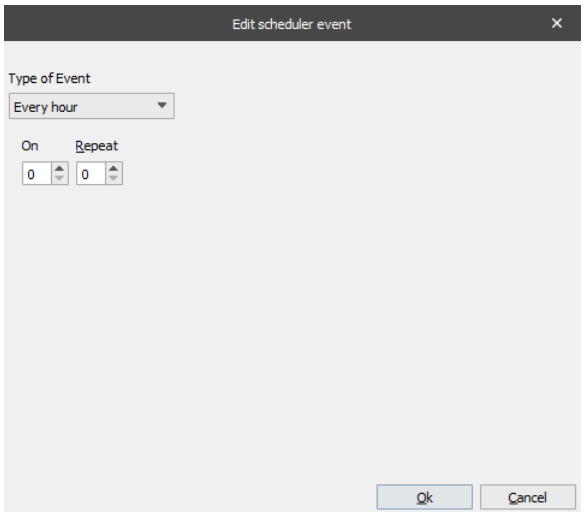
On the **Criteria** tab you can specify criteria that must be fulfilled for a job to be entered into the queue. If no criteria are fulfilled for a job, it is executed right away.



On the **Times** page, you specify when Lasernet empties the job queue. You can create one or more scheduler events here. Click **Add** to add a new scheduler event.

There are several ways to set up a scheduler event:

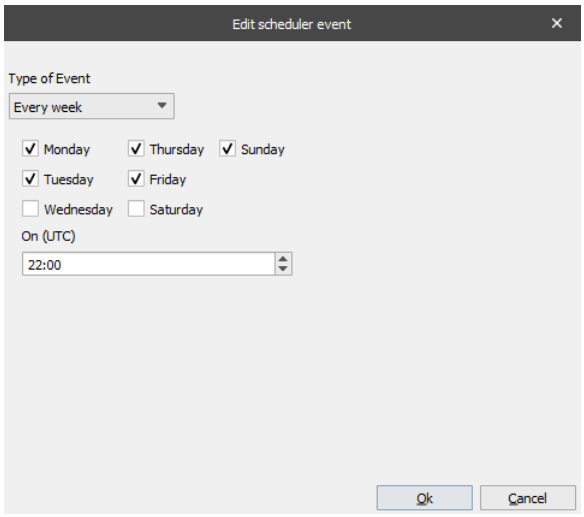
- Every hour
- Every week
- Every month on days
- Every month on weeks
- Every year
- Every minute



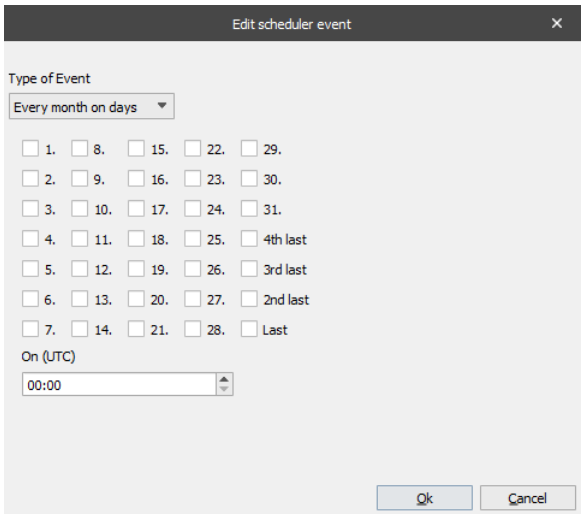
Every hour: At the start of every hour (according to the clock), Lasernet empties the queue after the number of minutes specified by **On**, and then repeatedly empties the queue at the interval (in minutes) specified by **Repeat** (until the end of the hour). This pattern is repeated every hour. If you set **On** to **0**, the queue is emptied at the start of the hour. If you set **Repeat** to **0**, the queue is emptied once every hour.

For example, if **On** is **30** and **Repeat** is **10**:

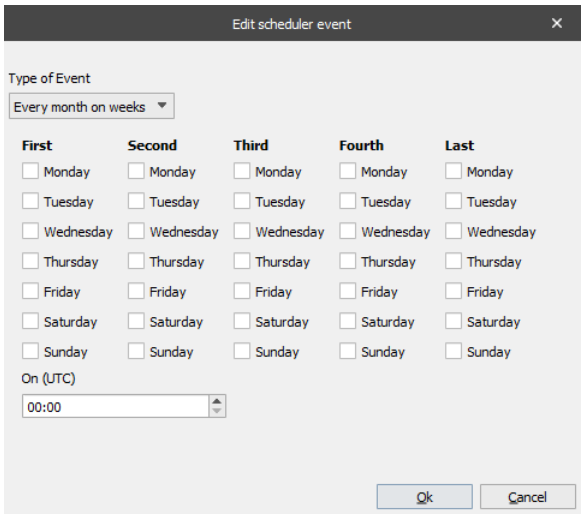
- A job queued at 11:00 is released at 11:30
- A job queued at 11:35 is released at 11:40
- A job queued at 11:55 is released at 12:30



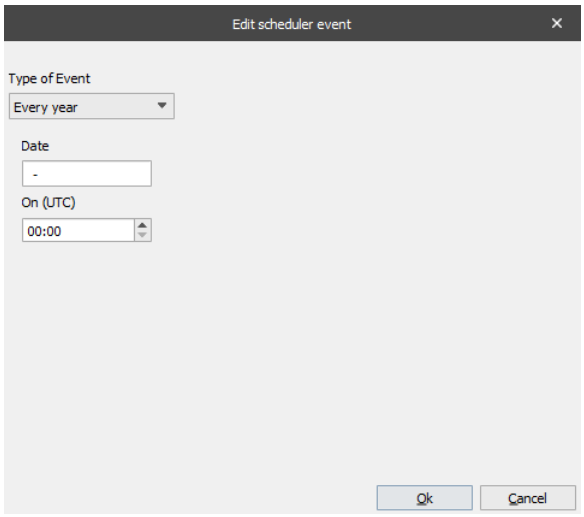
Every week: Empties the queue at a given time on given days.



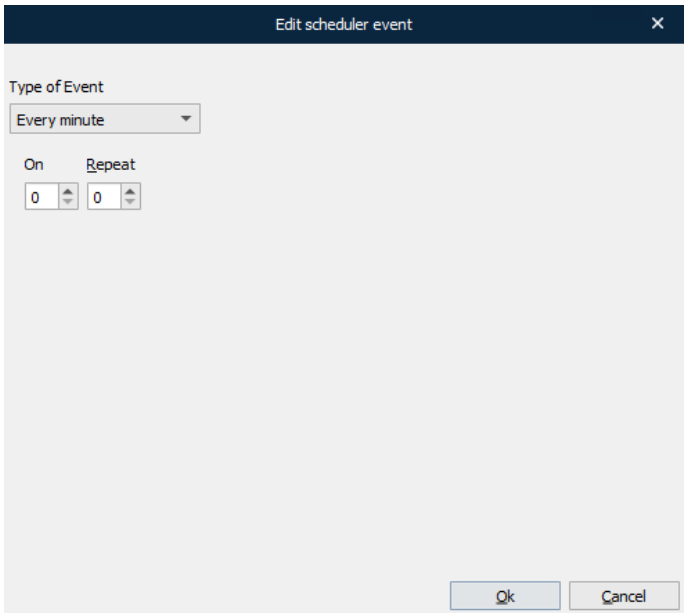
Every month on days: Empties the queue at the given time on the given days. The special day *Last* is the last day of the month, no matter how many days there are in that month. Likewise, with *2nd Last* etc.



Every month on weeks: Used for running on some specific weeks in the month. If you select **Sunday** in the **Second** week, the queue is emptied on the second Sunday in the month.



Every year: Enter a date in dd-mm format. The queue is emptied on that date.



Every minute: At the start of every minute (according to the clock), Lasernet empties the queue after the number of seconds specified by **On**, and then repeatedly empties the queue at the interval (in seconds) specified by **Repeat** (until the end of the minute). This pattern is repeated every minute. If you set **On** to **0**, the queue is emptied at the start of the minute. If you set **Repeat** to **0**, the queue is emptied once every minute.

For example, if **On** is **30** and **Repeat** is **10**:

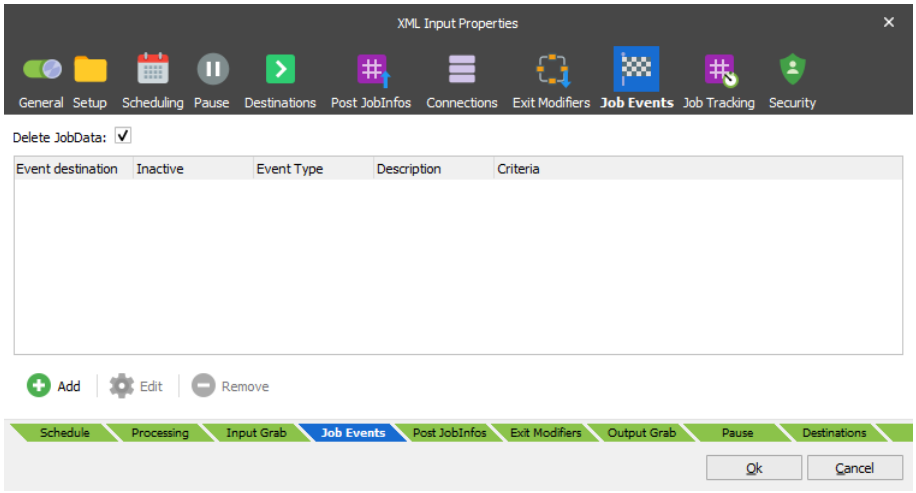
- A job queued at 11:00:00 is released at 11:00:30
- A job queued at 11:00:35 is released at 11:00:40
- A job queued at 11:00:55 is released at 11:01:30

Note: When you use **Every minute**, do not create a schedule that empties the queue so frequently that Lasernet becomes unresponsive.

13 Job Events.

13.1 Job Events

Job Events are available as a setting (see tab) on most modules.



They are used to generate an event when a Job reaches a predefined condition. A Job Event is effectively a new type of destination activated at a later time if certain criteria are met.

It is possible to have Lasernet activate when a Job or Job group completes, succeeds or fails.

Completion
Once Lasernet has finished processing the Job, the event occurs regardless of whether it failed or was successful.

Success
Once Lasernet has finished processing the Job, the event will only occur if the job was successful.

Failure
Once Lasernet has finished processing the Job, *the event will only occur if the job has failed*.

Levels
Jobs flow through Lasernet in a tree-like fashion. Each time a Job is passed, a clone or branch is created.

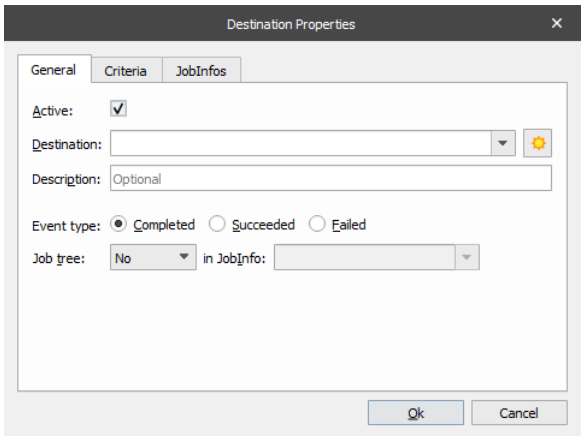
The level of the tree where the Job Event is set up determines the outcome. Only Jobs below this level are considered when determining if a Job group is completed, successful or failed.

13.2 JobData

When a Job Event exists, Lasernet stores the Job on disk for later use. When the event occurs, the Job is loaded from disk and sent to the Job Event destination. Hence it is the input Job and not the output Job of the tree which is used.

Generally, JobData is not needed for the event and as such it is not stored to disk or used (by default). However, it is possible to store JobData and use it in the event destination, if needed. This is done via the “Delete JobData” checkbox.

JobEvent
A JobEvent is defined by the following properties:



Destination	Where to send the Job and Job tree when the event occurs.
Event Type	What to monitor for; completion, success or failure.
Job tree	An optional XML document containing info about the Jobs and their status from the Job Event level in the tree and below. This tree can be put in any JobInfo and kept alongside the original JobData.

It is possible to have JobInfo criteria for determining if an event should occur or not.

A list of additional JobInfos to store in the XML Job tree is available.

Example

1. An input job is passed through an engine and out through an output module. A Job Event is defined for the input job.

When the event occurs, the Job is passed to the Job Event destination alongside the XML tree, which then contains info about these jobs:

- Input Job
- Engine Job
- Output Job

2. Instead of placing the Job Event on the input Job, it is put on the output Job. Now the tree will only contain info about the output Job and not the engine Job nor the input Job.

14 JobInfos.

14.1 JobInfos

14.1.1 Introduction

A JobInfo is a unique and variable information string, which is automatically generated in a module, or manually defined by a user. They are used as keywords and for setting up information in various modules. Generally, JobInfos are used for declaring an email address, an archive index or printer specific information. JobInfos can also be very useful for setting up conditions for validating the data stream. Lasernet will automatically build up a system list depending on the input module receiver, although user-defined JobInfos can be added as well.

JobInfos will be transferred between modules. This gives the ability to check, print, update and execute a current JobInfo at any statement, when processing a job through Lasernet. In turn, this ensures the seamless integration of Lasernet into a Windows environment.

To fully understand JobInfos, it is useful to first understand what constitutes a Job. Think of a Job as a package from the post office. It is a box with something in it – the raw data that you wish to process. In this analogy, JobInfos can be thought of as all the various labels that are attached to the package. One is the address where package is to be sent, another contains information about where the package originated and yet another shows the package has been processed by Customs en-route to its destination. Even the stamp is marked with a date that shows when the package was sent. All these extra pieces of key information attached to the package can be thought of as JobInfos. As well as containing *metadata* about the package, JobInfos also contain *metadata* about the Job. All JobInfos have a *name* which is used to access the data associated with it. In much the same way the Customs Label will be clearly marked ‘Customs’

The *name* of a JobInfo is *not* case-sensitive; the lines below are considered identical.

```
job.setJobInfo('a little jobinfo', 'With A Vengeance', true);
job.setJobInfo('A Little JobInfo', 'With A Vengeance', true);
```

Note: the value *is* case sensitive. This is important to remember when retrieving values and using them e.g. in comparisons.

14.1.2 JobInfo types

Most JobInfos can be represented as text, but sometimes the data type is such that it cannot easily be represented in this way. Internally, Lasernet handles all JobInfos (regardless of *type)* as binary data. In technical terms this means that when assigning a text to a JobInfo, Lasernet converts it to UTF-8 (a Unicode representation that can be stored byte-wise). When using JobInfos as text (the most common usage), this does not pose any problems. However, a few situations require special attention. This includes the File Output module, which has the ability to embed JobInfos into the JobData. In this instance it is possible to choose which codepage the JobInfo should be converted to before embedding it.

14.1.3 JobInfo lists

In some situations, it is necessary or desirable to have more than one value in a JobInfo. For instance, the *Source* JobInfo is a *list* of modules that the Job has passed through; a new value being added to the *Source* JobInfo for every new process. The function “Replace existing JobInfo”, commonly found throughout Lasernet, can knock out these accumulated values and replace them with a single user defined value instead. If “Replace existing JobInfo” is not activated, any existing values will be preserved.

The individual items can then be accessed in different ways. If there is a JobInfo list called MyJobInfo, accessing the third element of the list can be done with a script function such as getJobInfo("MyJobInfo",2) or by creating a Fixed Text Rearrange of type JobInfo and writing MyJobInfo[2]. The indexing is zero-based, meaning that the first item of the list is MyJobInfo[0].

14.1.4 JobInfo Substitution

Since the JobInfos are an essential part of a Job it is possible to assign values to them in many different ways and at multiple locations. When assigning a value to a JobInfo you must specify the *name* of the JobInfo as well. The value does not have to be text. Special syntax exists which allows for greater functionality. This is generally known as *substitution*. For example, this allows you to write:

```
The job came from #InputPort#
```

as a *value*. This means that Lasernet will look up the value in the *InputPort* JobInfo and replace *#InputPort#* with that value. This can be done with more than one JobInfo:

```
The job came from #InputPort# and was created at #JobTime#
```

The *#JobInfoName#* syntax has a few more possibilities. Since a JobInfo can also be a list, you can choose to access a specific value in the list, by adding an *index* to the substitution:

```
The job came from #Source[0]#
```

You can also choose to expand the list with a custom delimiter:

```
The job has passed through these modules: #Source{,}#
```

To facilitate the use of # as a value, an escape character is used – The Generic Currency Symbol – ¤ – Ascii A4.

Putting a ¤ before the # allows you to use # as a value. To get a ¤ as value, simply use two: ¤¤

If you only need one # simply put that in. As long as there is no matching # Lasernet assumes you just want the one #

In the following samples |NAME| means the contents of the JobInfo with the name.

```
abc¤###def => abc##def
abc¤¤#def  => abc¤#def
 abc¤¤#def# => abc¤|def|
```

JobInfo substitution is not recursive, so if a JobInfo name is put in a JobInfo value, only the outer JobInfo is substituted e.g., if there are two JobInfos: Inner=Value, Outer=#Inner#, typing "#Inner#" in a field with substitution will yield Value, while putting "#Outer#" will yield "#Inner", which will not be substituted further.

Due to historical implementation issues, there may be some places in Lasernet where this approach does not work.

14.1.5 JobData as JobInfo

As JobInfos are incredibly versatile the actual raw job data is also stored in a JobInfo. This JobInfo is called *JobData* (by default).

This means that you can change the value of the raw data by using normal substitution expressions as detailed above. Caution should be exercised when using this approach as the substitution is not foolproof,

especially when working with a mixture of binary and text data. Whilst results are generally favorable, unexpected output may occur. As such, this function should only be used after careful consideration.

14.1.6 System generated JobInfos

JobInfos are automatically generated by Lasernet. This information is generated by the specific input modules which receives the data.

Dynamic JobInfos give quick and easy access to key information:

JobInfo Name	Description
JobSize	The size of the Job (in bytes)
ConfigRevision	To get the revision of the configuration on a form. In the Developer app returns 'Patched' unless an older revision is opened. Lasernet Service returns the deployed revision number.
CurrentTime	Always returns the current time as hh:mm:ss.zzz (where *zzz* is milliseconds)
CurrentDate	Always returns the current date as yyyy-MM-dd
LocalHost	Returns the name of the current computer
CurrentSubmoduleID	The name of the module currently processing the Job.
JobID	A unique ID for the Job – this can change for each sub module that the Job passes through
JobData	The standard JobInfo containing the raw data to process
PrimaryJobData	The name of the JobInfo that contains the JobData – by default the value is JobData (NB: This JobInfo is deprecated and will be removed in future releases)
PublicID	The JobID of the Job when it was first created.
CurrentModuleID	The full target URL of the module that is currently handling the Job
OriginatingModule	The module that created the Job – usually the same as Input module but could also be an instance of the Forms Engine that creates and manipulates copies of the Job
InputPort	The Input module that created the first instance of the Job
JobTime	The time the Job was started
JobDate	The date the Job was started
Source	A list of the modules that the Job has passed through

14.1.7 Manually generated JobInfo in the ERP-system

A JobInfo can be declared in any Windows application, such as an ERP-system. Simply type the JobInfo string as pure text. The syntax for manually setting up a JobInfo in a Windows application is:

#JobInfo JobInfoType=JobInfoValue# like

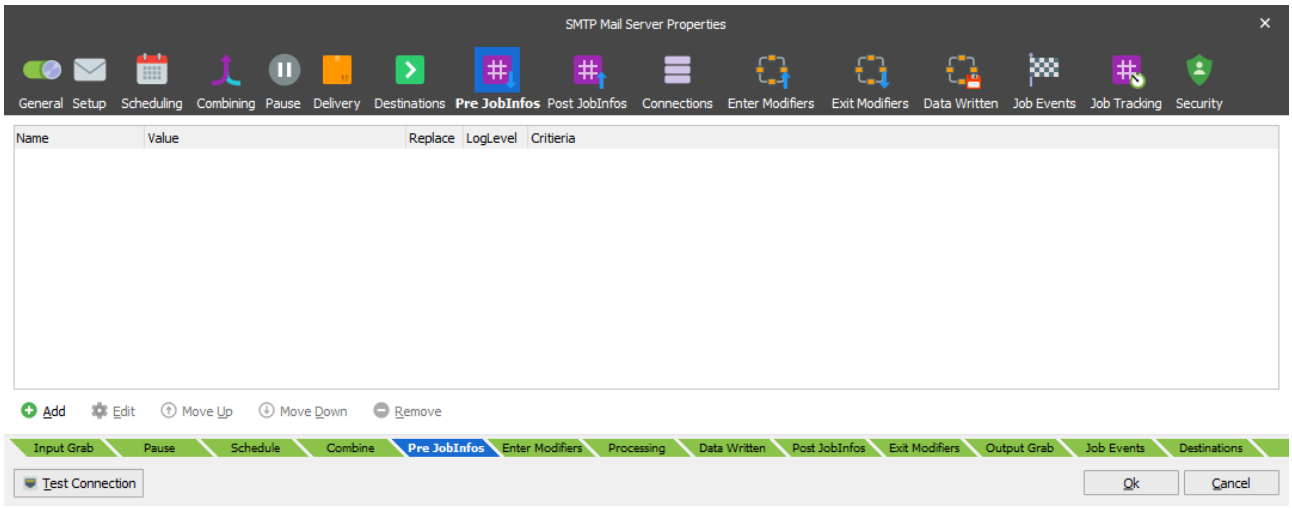
```
#JobInfo MailTo=john.doe@hotmail.com# or
#JobInfo MailTo=john.doe@hotmail.com; jane.doe@hotmail.com#
```

(comma and semicolon are valid as separator characters for multiple addresses)

The JobInfo text can be defined with any font, size or typeface and Lasernet will still scan the spool job and pick up the keywords.

14.1.8 JobInfo defined via UI in modules

A JobInfo can be defined in any module at different event types, called **Pre JobInfos** and **Post JobInfos**, depending on the module type. The green tabs at the bottom of the module dialogs illustrate the processing order of jobs.



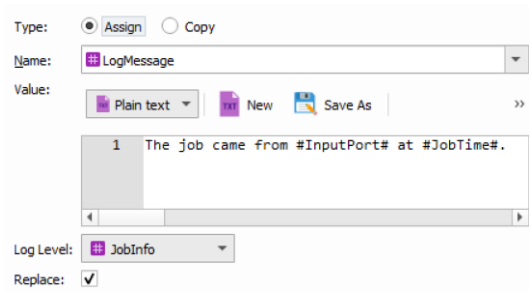
To set up a JobInfo in a Lasernet module, fill out the JobInfo Name and JobInfo Value by clicking the **Add** button. In some cases, you will find a button named **Add Copy JobInfos**. This button is used to copy the contents of **JobData** in Binary mode to **PreviewJobData**. This used to preview the original contents of the input data in Lasernet Client.

Type: Select **Assign** to assign a value in text mode. Select **Copy** to copy a value in binary mode.

Name: Includes a list of the most commonly used, pre-defined Lasernet JobInfos, or a custom named JobInfo.

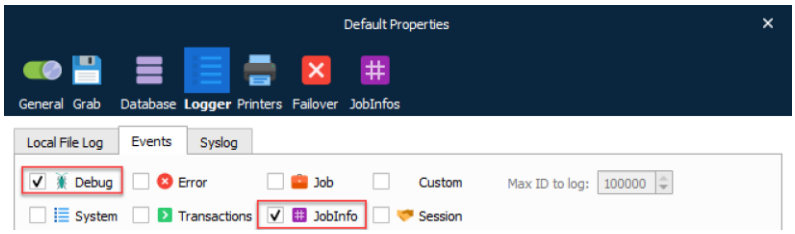
Value/Copy: This field is empty by default. However, several JobInfos include a list of typical values to choose from. These are only provided for guidance and do not have to be used.

In **Assign** mode the value can be set as a mix of a JobInfo and clear text, like



In **Copy** mode a binary value is copy from the name of JobInfo to the name of another JobInfo:

Log Level: Used to maintain the status of the server log message. Default value is **JobInfo** level that will show the name and value of the log message. **Debug** mode is only show the name and value in debug mode and log level **NoLog** will not create a log message.



Replace: By activating this checkbox is will replace any other JobInfo with the same name. Some JobInfos can contain a list of several values for example, an email distribution list.

The ***Move up*** and ***Move down*** buttons are to re-order in which the JobInfos are defined.

14.1.9 JobInfo Reference

JobInfo Name	Description
ActiveCodePage	Lasernet attempts to keep this JobInfo up to date with the code page of any text currently residing in JobData. It is set by the Code Page Conversion Modifier, the Text Filter Modifier and the Filter Engine. The Code Page Conversion Modifier sets it to the name of the output codec. Since the Text Filter Modifier and the Filter Engine have options for disabling code page conversion, they both check on the value of the ActiveCodePage JobInfo. If it exists, they convert from the codec it suggests to UTF16, filter the data and then convert to the desired output format. If no output format is chosen it defaults to UTF8. The value of ActiveCodePage is then set to the name of the output codec (default UTF8).
ActualOutputFilename	The ActualOutputFilename JobInfo is created by the File Output module. The value is set to the name of the output file actually written to disk. The filename includes the entire path. The value could be used in a Data Written Modifier.
ArchiveFileName	Contains the name of the archive file. Created by the Compression module if decompress is enabled.
Autofill	Created by Meta Input. Available when an Autofill request has been sent from Lasernet Meta. The value is equal 1.
AutoPaperSource	Set value to "1", "Yes" or "True" to automatically select the paper source in the printer, by detecting the PDF page size, for PDF documents stored in the JobInfo PrintAttachment. Modules: Printer Output, Printer Service
CleanJobInfos	Used by the Database Command. If set to 1, JobInfo Substitution will not occur on the JobInfos whose assigned values are returned from the database. If set to anything else they will be substituted.
ClientID	ClientID is used together with PreviewExtension to give the client further information.
ClientUserName	Created by Meta Input. Username for the login client as defined in the Lasernet Meta → Tools → Client settings. When running with default credentials the value will be left empty.
ColorMode	Supported by EMF2RAW, Printer Output and Printer Service to set monochrome or color printing.
Copies	Supported by Printer Output and Printer Service to control the number of printed copies.
CompressionLevel	Contains the compression level of the zip archive. Created by the Compression module if decompress is enabled.
CurrentDate	The CurrentDate JobInfo is always available. It contains the current date in the format yyyy-MM-dd. The date is fetched from the computer on which Lasernet is running.
CurrentModuleID	The CurrentModuleID JobInfo contains the full path to the module in which the job currently resides.

CurrentPage	The CurrentPage JobInfo is set in the Form Engine when the analysis has finished fitting text to the pages and has started running scripts on rearranges. It is set to the page number on which the script for rearranges is currently being run.
CurrentTime	The CurrentTime JobInfo is always available. It contains the current time in the format hh:mm:ss.zzz where zzz represents milliseconds. The time is fetched from the computer on which Lasernet is running.
DataFormat	The DataFormat JobInfo is created by the Printer Input module. The value is the data format generated by the printer that the engine is using. If the DataFormat is empty, the Printer Input module sets it to the default value EMF. It is also created by the PDF Modifier where it is consistently set to the value RAW. It is also created by the Form Engine where it can be set to TEXT or EMF. DataFormat is used by EMF2RAW.
DataParameters	Created by the Printer Input module. Contains parameters from the print processor which handled the job.
DataPrinterName	Created by the Printer Input module. Contains the name of the printer that printed the job.
DataStartTime	Created by the Printer Input module.
DataTotalPages	Created by the Printer Input module. Contains the total number of pages in the print job. May be zero if the job does not contain any page delimiting information.
DataUntilTime	Created by the Printer Input module.
Default, DefaultPrinter	The Default or DefaultPrinter JobInfos are options that can be set by several input modules. This value can be used to store a destination that can be used later, for example, in the form engine. This is convenient for pairing input and output modules but letting the job pass through the same engines somewhere in the middle.
Destination	Will be set each time a job is passed to another engine or an output module. It is set to the name of the receiver.
DetailInformation	Created by various modules. It is primarily used by the Lasernet Monitor to show module specific information about the job. It is possible for the user to add information to this JobInfo but be aware that it may be overwritten at any time.
DocName	Created by the Printer Input module. It contains the name of the print job as set by the Windows Print Spooler. If it is empty the default value set by Printer Input module is *Unnamed – Lasernet document*. DocName also is supported in the Printer Output, Printer Service modules to set name of print job for the Windows printer queue.
DocumentName	Created by Meta Input. The name of the print job, as set by the Windows Print Spooler, when the document is printed to Lasernet Meta. Or the name of the file, as set by the Windows File System, when the document is retrieved via polling in Lasernet Meta.
DropboxURL	URL to publicly shared file. The JobInfo is only set when **Sharing publicly** is enabled in the GUI.
DriverName	The name of printer driver as set by the Windows Print Spooler, when the document is sent to Meta Input (via Lasernet Meta) or direct to the Printer Input module.
DuplexMode	Allows to print on both sides of a paper automatically. Examples of values are: Default, Simplex, Vertical and Horizontal. Supported by EMF2RAW, Printer Output and Printer Service modules.
Extension	Created by the Azure Storage Input, File, FTP, HTTP and Meta Input modules. The value is set to the extension of the file which is received by module e.g., '.txt'. The extension includes the dot (.) in front.
ExitCode	Is generated by the Process Modifier. It contains the actual exit code of the external application.
FailingPort	The FailingPort JobInfo is set when a job is failed.
FailingSubmodule	The FailingSubmodule JobInfo is set when a job is failed. It is set to the name of the submodule in which the job fails.

FileCreated	Set by File Input module to the time the file was created. The time is in the format *yyyy-mm-dd hh:mm:ss.zzz*.
FileComment	Contains a comment embedded in the zip archive if available. Created by the Compression module if decompress is enabled.
FileID	ID of uploaded file returned after upload to Google Drive or OneDrive via the output modules.
FileLastModified	Set by File Input or Compression module to the last modified time of the incoming file (for Compression module file(s) inside the archive). The time is in the format *yyyy-mm-dd hh:mm:ss.zzz*.
FileLastRead	Set by File Input module to the last read time of the file. The date is in the format *yyyy-mm-dd hh:mm:ss.zzz*.
Filename	The Filename JobInfo is created by the File Input module. The value is set to the filename (without the path – see Filepath) of the file which is picked up by File Input module. Also set by the Azure Storage Input, FTP and HTTP modules to the name of the file (without URL) retrieved from the server. Filename is used by the File Output module to specify where to save the job data. Filename is also used by the Mail Output module to set the filename of optional attachments. Filename is also used by the PDM Output module to specify the file to be archived.
FileNameWithoutExt	The FilenameWithoutExt JobInfo is created by the File Input module. The value is set to the name of the file which is picked up by File Input module, excluding the path, extension and the dot (.). If the filename picked up is called myfile.txt then FilenameWithoutExt is set to myfile. Also set by the Azure Storage Input, FTP and HTTP modules according to the same rules.
FilePath	The FilePath JobInfo is created by the File Input module. The value is set to the path (excluding filename) where the file is picked up by the File Input module. Also set by the Exchange, FTP and Mail Input modules. Here it is the URL of the file without filename.
FileRelativePath	The relative path to the root of the file is created by the File Input module.
FileRootPath	The path to the root of the file without filename is created by the File Input module.
FileSize	The FileSize JobInfo is created by the File Input module. The value is set to the size of the file which is picked up by the File Input module. If using the JobInfo Scanner feature, the FileSize JobInfo is not set as the actual data processed may be in one or more files. Instead, use the JobSize JobInfo which is set to the size (in bytes) of the job actually being processed Set by the Exchange, FTP, HTTP and Mail Input modules to the size of the file retrieved from the server. Also set by the Azure Storage Input module.
FileSizeCompressed	Contains the file size of the compressed archive file. Created by the Compression module if decompress is enabled.
FirstPageInJob	The value will be set to 1 (boolean) when the Form module processes the first page in forms running in page to job mode. Otherwise, it is set to 0 (zero).
FirstPageInSpoolJob	The value will be set to 1 (boolean) when the Form module processes the first page in a spool job.
FJobDate	The FJobDate JobInfo is always available. It contains the date on which the job was created. The time is in the format yyyy_MM_dd. The corresponding time is in the JobInfo JobTime. A more user-friendly version of this JobInfo is JobDate.
FJobTime	The FJobTime JobInfo is always available. It contains the time at which the job was created. The time is in the format hh_mm_ss_zzz. The corresponding date is in the JobInfo FJobDate. A more user-friendly version of this JobInfo is JobTime.
FormName	FormName is used by EMF2RAW and Printer Output module to set a paper format supported by the printer driver.

Formtype	Set Formtype to define a custom queue name in Lasernet Client for paused, scheduled or failed jobs. Name of queue will figure as a child node to the module for where the job is being saved. If Formtype is not defined, for a job, an empty name is listed in Lasernet Client as the queue name.
FQDN	Created by Meta Input. Fully qualified domain name for the user running Lasernet Meta.
FullFilename	The FullFilename JobInfo is created by the File Input module and Azure Storage Input module. The value is set to the full filename – including path, filename and extension (see Filepath and Filename) of the file which is picked up by the File Input module.
GrabWidth	The GrabWidth JobInfo is set by the Form Engine. The value is set to the maximum width of the grab being handled by the recognized form.
InputBody	Set by the Exchange and Mail Input modules to the body of the mail, if any, of an incoming email.
HTTPStatusCode	The HTTP status code received by the HTTP modifier or output module.
HTTPStatusText	The HTTP status text received by the HTTP modifier or output module.
HTTPHeaderFieldName HTTPHeaderFieldValue	A pair of JobInfo lists that contain the names and values of the HTTP headers received by the HTTP modifier or output module. Each header's name is an item in HTTPHeaderFieldName. That header's value is the corresponding item in HTTPHeaderFieldValue. For example, if `HTTPHeaderFieldName[5]` = `Content-Length`, the value of the `Content-Length` header (in this example, `28`) is in `HTTPHeaderFieldValue[5]`. Each header is also stored in a dedicated JobInfo (see HTTPHeader* below).
HTTPHeader*	Each response header received by the HTTP modifier or output module is stored in a dedicated JobInfo named HTTPHeader<header name>. For example, if the value of the `Content-Length` response header is `28`, Lasernet creates a `HTTPHeaderContent-Length` JobInfo and gives it the value `28`.
InputBodyHTML	The HTML contents of the mail body, if any, of an incoming email.
InputDefaultSource	Set by the Print Input module to the value of the selected default source for the incoming document. If available, the value is presented as a number.
InputDuplexMode	Set by the Print Input module to the value of the selected duplex mode for the incoming document. Known values are Simplex, Horizontal and Vertical.
InputFilename	Set by the Exchange, Mail Input and Outlook Mail input modules to the filename of the attachment.
InputFromEmail	Set by the Exchange and Mail Input modules to the email address of the sender.
InputFromName	Set by the Exchange and Mail Input modules to the display name of the sender.
InputHeaderFieldName InputHeaderFieldValue	A pair of JobInfo lists that contain the names and values of the custom (X- or x-prefixed) mail headers received by the Mail Input module and Outlook Mail input module. Each header's name is an item in InputHeaderFieldName. That header's value is the corresponding item in InputHeaderFieldValue. For example, if InputHeaderFieldName[5] = X-Priority, the value of the X-Priority mail header (in this example, 3) is in InputHeaderFieldValue[5].
InputHeaders	Set by the Exchange module and contains transport-specific message envelope information if available.
InputLongFilename	Set by the Mail Input module to the long filename of the attachment. Also set by the FTP and HTTP Input modules to the full name of the file – URL + filename.
InputMessageID	Set by the Exchange and Mail Input modules to an unique identifier for the incoming email
InputMimeType	Set by the Exchange and Mail Input modules to the MIME type of the attachment.
InputOrientation	Set by the Printer Input module as the Paper orientation for the incoming print document as created by the Lasernet EMF driver.

InputPaperHeight	Set by the Printer Input module as the length of the paper for the incoming print document in tenths of a millimetre as created by the Lasernet EMF driver. Margin size is not included.
InputPaperSize	Set by the Printer Input module as the paper size for the incoming print document as created by the Lasernet EMF driver. Paper size is represented as a unique number (like A4 = 9)
InputPaperWidth	Set by the Printer Input module as the width of the paper for the incoming print document in tenths of a millimetre as created by the Lasernet EMF driver. Margin size is not included.
InputPort	The InputPort JobInfo contains the full path to the module in which the job was originally created. That is usually the name of the Input module. The Form Engine uses the InputPort JobInfo in Page To Job mode. It is used for grouping the pages from a Job into the right queues. In theory you could manipulate the InputPort JobInfo to force pages into the same or different queues.
InputPrintQuality	Set by the Printer Input module as the print quality for the incoming print document as created by the Lasernet EMF driver. Value always contains a number set to 1.
InputScale	Set by the Printer Input module as the print quality for the incoming print document as created by the Lasernet EMF driver. Value always contains a number set to 100
InputSubject	Set by the Exchange and Mail Input modules to the subject of the mail.
InTempFilename	Created by the Process Modifier. Contains the name of the temporary file (if any) created with data from the external application.
JobDate	The JobDate JobInfo is always available. It contains the date on which the job was created. The date is in the format yyyy-MM-dd. The corresponding time is in the JobInfo JobTime. A more filename friendly version of this JobInfo is FJobDate.
JobID	The JobID JobInfo contains a fully unique ID for the job. The format of the JobID may change between Lasernet versions, so do not rely on it.
JobSize	The JobSize JobInfo is always available. It contains the size of the primary JobData (in bytes). Usually stored in the JobData JobInfo.
JobTime	The JobTime JobInfo is always available. It contains the time at which the job was created. The time is in the format hh:mm:ss.zzz. The corresponding date is in the JobDate JobInfo. A filename friendly version of this JobInfo is FJobTime.
LastPageInJob	The value will be set to 1 (boolean) when the Form module processes the last page in forms running in page to job mode. Otherwise, it is set to 0 (zero).
LastPageInSpoolJob	Contains a Boolean that is set to true (1) by the Form module when last page in spool job is being processed. Otherwise, it is set to 0 (zero).
LocalHost	The LocalHost JobInfo is a JobInfo created by system, that are available everywhere. It is automatically set to the name of the current computer.
MachineName	Created by Meta Input and Printer Input. The name of the machine on which the job was printed from. Example of machine name formats: **Lasernet** (Meta Input) or **\\Lasernet** (Printer Input).
MailAttachment	The Exchange, Outlook Mail and Mail Output modules use the MailAttachment JobInfo array to add attachments to the email being sent. One attachment is added for each entry in this array. The following JobInfos are used together with this JobInfo: • MailAttachmentFilename • MailAttachmentMimeEncoding • MailAttachmentMimeType

MailAttachmentFilename	The Exchange, Outlook Mail and Mail Output modules use the MailAttachmentFilename JobInfo array to name the attachments added to the email being sent. The following JobInfos are used together with this JobInfo: MailAttachment MailAttachmentMimeEncoding, MailAttachmentMimeType.
MailAttachmentMimeEncoding	The Exchange, Outlook Mail and Mail Output modules use the MailAttachmentMimeEncoding JobInfo array to set the mime encoding of the attachments added to the email being sent. If this JobInfo does not contain a corresponding entry for a MailAttachment entry, a default of base64 is used. The following JobInfos are used together with this JobInfo: • MailAttachment • MailAttachmentFilename • MailAttachmentMimeType
MailAttachmentMimeType	The Outlook Mail and Mail Output modules use the MailAttachmentMimeType JobInfo array to set the mime type of the attachments added to the email being sent. If this JobInfo does not contain a corresponding entry for a MailAttachment entry, a default of text/plain is used. The following JobInfos are used together with this JobInfo: • MailAttachmentFile • MailAttachmentFilename • MailAttachmentMimeEncoding.
MailBCC	The Exchange, Outlook Mail and Mail Output modules use the MailBCC JobInfo to set the bcc mail recipient. This is typically the actual email address in the standard internet format: name@domain.com. Comma and semicolon are valid as separator characters if MailBCC contains multiple addresses.
MailBCCName	The Exchange, Outlook Mail and Mail Output modules use the MailBCCName JobInfo to set the Display Name of the bcc mail recipient.
MailBody	The Exchange, Outlook Mail and Mail Output modules use the MailBody JobInfo to set the actual contents of the mail. Please note that if you want to send html-email you should use MailBodyHTML instead.
MailBodyHTML	The Exchange, Outlook Mail and Mail Output modules use the MailBodyHTML JobInfo to set the actual contents of the mail when sending HTML-email. A standard text email should use the MailBody JobInfo instead.
MailBodyMimeType	Discontinued.
MailCC	The Exchange, Outlook Mail and Mail Output Modules use the MailCC JobInfo to set the cc mail recipient. This is typically the actual email address in the standard internet format: name@domain.com. Comma and semicolon are valid as separator characters if MailCC contains multiple addresses.
MailCCName	The Exchange, Outlook Mail and Mail Output modules use the MailCCName JobInfo to set the Display Name of the cc mail recipient e.g., John Doe.
MailDraft	Use the MailDraft JobInfo as a Boolean to overwrite the setting "Create draft without sending it" in the Outlook Mail Output module. • 0 = Do not create a draft before sending email • 1 = Create a draft before sending email
MailFolder	The Mail Input and Outlook Mail modules use the MailFolder JobInfo.
MailFrom	The Exchange, Outlook Mail and Mail Output modules use the MailFrom JobInfo to set the address of the mail sender. This is typically the actual email address in the standard internet format: name@domain.com.

MailFromName	The Exchange, Outlook Mail and Mail Output modules use the MailFromName JobInfo to set the Display Name of the sender
MailJobData	The Exchange, Outlook Mail and Mail Output modules use the MailJobData as a Boolean to overwrite the setting “Do not insert job data in an attachment” which is available as a setting in the module. • 0 = Do not insert job data in an attachment • 1 = Insert job data in attachment
MailImportance	The Outlook Mail module uses this JobInfo. • 0 = Low • 1 = Normal • 2 = High
MailMessageID	A unique identifier created after a successful delivery to the SMTP server via the Exchange, Outlook Mail and Mail Output modules. If a connection cannot be established the JobInfo will not be created.
MailReplyTo	Responses to messages you send with an alternative MailReplyTo address, via the Exchange, Outlook Mail and Mail Output modules, are delivered to that address. Your MailTo address will still appears in the From field.
MailRequestDeliveryReceipt	The Outlook Mail module uses this JobInfo. At runtime, it will overrule the request delivery receipt selection in the module settings. When set to true, a receipt is returned upon successful delivery to the recipient’s mailbox. • 0 = Do not send back • 1 = Send back
MailRequestReadReceipt	The Outlook Mail module uses this JobInfo. At runtime, it will overrule the request delivery receipt selection in the module settings. When set to true, the receiver of the mail is asked to send back a receipt upon receiving the mail. • 0 = Do not send back • 1 = Send back
MailSubject	The Exchange, Outlook Mail and Mail Output modules use the MailSubject JobInfo to set the subject of the mail.
MailTo	The Exchange, Outlook Mail and Mail Output modules use the MailTo JobInfo to set the mail recipient. This is typically the actual email address in the standard internet format: name@domain.com. Comma and semicolon are valid as separator characters if MailTo contains multiple addresses.
MailToName	The Exchange, Outlook Mail and Mail Output modules use the MailToName JobInfo to set the Display Name of the mail recipient e.g., John Doe.
MatchedMask	The MatchedMask JobInfo is created by the File, FTP and HTTP Input modules. The value is set to the mask used when a file is picked up by the module e.g., ‘*.txt’.
MD5FileName	Used to override the default name of the pre-uploaded checksum file. The ".md5" file extension indicates a checksum file containing 128-bit MD5 hashes in md5sum format.
MetaQueue	Created by Meta Input. The name of the meta queue from which the document is sent in Lasernet Meta.
MimeEncoding	MimeEncoding is used by the Mail Output modules to set the MimeEncoding of an optional attachment.

MimeType	The Mail Output module uses the MimeType JobInfo to set the mime type of an optional attachment.
ModifierErrorMessage	Please see ModifierFailed.
ModifierFailed	An error can be tracked as a failed modifier and will set the JobInfo *ModifierFailed* to 1 (true). The JobInfo *ModifierErrorMessage* is set to the (internal) error message that caused the failure. For a list of modifiers, it is possible to add a criterion that will only run them if ModifierFailed is not 1 (true).
MSMQLabel	Label of the MSMQ message. NB: Cannot be longer than 250 characters.
NotifyName	Created by the Printer Input module. Contains the name of the user that should be notified about print progress.
NumberOfPages	The NumberOfPages JobInfo is set in the Form Engine when the analysis has finished fitting the text to the page(s). It is set to the number of pages that the analysis has created.
NumInputPages	Total number of physical pages in the EMF spool job as received by the Lasernet EMF driver on the Printer Input module.
OCRAutoCapture	If OCRAutoCapture is equal 1 (true) the AutoCapture process was running in the OCR Engine and has created a temporary OCR Form. To differ from OCRMatch it will be true in both standard and AutoCapture OCR Forms.
OCRMatch	If OCRMatch is equal to 0 (false) the job is not recognized by an OCR Form in the OCR Engine.
OCRValidated	If OCRValidated is equal to 0 (false) the job is not validated successfully after being processed by the OCR Engine.
OffsetX	The OffsetX JobInfo is set by the Printer Input module. It contains the offset of the EMF print from the side of the page.
OffsetY	The OffsetY JobInfo is set by the Printer Input module. It contains the offset of the EMF print from the top of the page.
Orientation	Manage the orientation of the paper in EMF2RAW and Printer Output modules. Valid values are Portrait or Landscape.
OneDriveSharePublicly	Set the OneDriveSharePublicly JobInfo as a Boolean value to overwrite the setting for Share publicly property.
OneDriveURL	URL to publicly shared file, returned after upload. The JobInfo is only set when sharing publicly is enabled in the GUI.
OneDriveURLExpiration	This OneDriveURLExpiration JobInfo overwrites the offset of the Shared Link Expiration defined in the OneDrive module. An expiration time for a shared document must be defined in the format yyyy-MM-ddTHH:mm:ssZ.
Orientation	Manage the orientation of the paper. Example of values are Portrait or Landscape. Supported by EMF2RAW, Printer Output and Printer Service modules.
OriginatingModule	The OriginatingModule JobInfo contains the full path to the module in which the job was originally created. This is usually the name of the Input modules.
OutTempFilename	Created by the Process Modifier. Contains the name of the temporary file (if any) created with data from the external application.
PageSeparators	The PageSeparators JobInfo is created by the File Input module. It matches the value of the page separators entered in the setup when using the JobInfo Scanner feature.
PageWidth	The PageWidth JobInfo is set by the Form Engine. The value is set to the maximum width of the page currently being processed.
PagesInSpoolJob	Number of input pages in the spool job as detected by the Form module when running in Text input mode.

PagesCombined	The PagesCombined JobInfo is set by the Form Engine. The value is set to the number of pages in the whole job being handled by the form. This is usually only relevant in connection with Page to Job forms.
PaperHeight	PaperHeight defines the height of the paper form. Supported by EMF2RAW, Printer Output and Printer Service modules.
PaperSource	PaperSource selects the Paper Source in the printer. Example of values are: Auto, Tray 1, Tray 2, Upper Tray, Lower Tray. Supported by EMF2RAW, Printer Output and Printer Service modules.
PaperWidth	PaperWidth defines the width of the paper form. Supported by EMF2RAW, Printer Output and Printer Service modules.
PDMATT	Specifies only metadata is being sent. Possible values are “Yes” and “No”.
PDFDecrypted	Result of decryption check in the PDF Security Module. 1 if successfully decrypted, o (zero) if unsuccessfully decrypted.
PDFEmbedAdditionalMetadataElement	Used by the PDF module and allows you to embed any number of additional XML metadata elements in the PDF by creating an array. Each entry must be valid XML.
PDFEmbedDescription	Used by the PDF module. This JobInfo contains a description of the embedded file. The field is optional.
PDFEmbedData	Used by the PDF module. This JobInfo must contain the (binary) content of the file to embed into the PDF.
PDFEmbedFilename	Used by the PDF module. Defines the name of the embedded file inside the PDF.
PDFEmbedPDFAExtensionSchema	Used by the PDF module for PDF/A Extension Schemas. Any number of schemas can be embedded by creating an array. Each entry must be valid XML.
PDFEmbedRelationship	Used by the PDF module. This field must be set to either *Source*, *Data*, *Alternative* or *Supplement*: The standard describes which value to use, depending on the embedded files relation to the PDF: **Source** shall be used if this file specification is the original source material for the associated content. **Data** shall be used if this file specification represents information used to derive a visual presentation – such as for a table or a graph. **Alternative** shall be used if this file specification is an alternative representation of content, for example audio. **Supplement** shall be used if this file specification is a supplemental representation of the original source or data that may be more easily consumable (e.g., A MathML version of an equation). If no PDFEmbedRelationship is defined, or it is set to an invalid value, this will default to *Supplement*.
PDFEmbedSubType	Used by the PDF module. This JobInfo must contain the MIME type of the embedded file. If not specified it will default to *application/octet-stream*.
PDFOwnerPassword	Used by the PDF Security module to set the owner password for an encrypted PDF file. The value will overwrite the defined owner password in the PDF Security module at runtime.
PDFSignatureValid	Result of validation in the PDF Security Module. 1 if valid, 0 (zero) if not valid.
PDFSigned	Result of signing check in the PDF Security Module. 1 if signed, 0 (zero) if not signed.
	Reserved
PDFUaStructureTree	JSON that describes the structure tree for PDF/UA documents.
PDFUserPassword	Used by the PDF Security module to set the user password for an encrypted PDF file. Value will overwrite the defined user password in the PDF Security module at runtime.
PDMCUK	Unique customer identification.

PDMDFMT	Datetime format.
PDMDKEY1-15	Up to 15 datetime keys can be specified. The format is defined in the module.
PDMDOC	Name of the document definition.
PDMLINE	JobInfos starting with PDMLINE are read as extended keys. Extended keys are setup in the archive and the data type must be used accordingly. An example of an extended key: PDMLINEaccountnumber=12345
PDMNKEY1-15	Up to 15 numeric keys can be specified. The format is defined in the module.
PDMSKEY1-25	Up to 25 string keys can be specified.
PDMUPD	Specifies if an existing document should be updated. Possible values are “Yes” and “No”.
PreviewExtension	Created by Meta Input in Autofill mode or added by clicking “Add Copy JobInfo” button when adding a new JobInfo in Lasernet Developer (see also PreviewJobData). PreviewExtension contains the filename extension to be used for the preview module. When a destination is set to preview, Lasernet tries to route the job back to the client for preview. The PreviewExtension then tells the client how to do the preview. If no PreviewExtension has been set, the default value is .pdf.
PreviewJobData	Created by Meta Input in Autofill mode or added by clicking “Add Copy JobInfo” button when adding a new JobInfo in Lasernet Developer. PreviewJobData contains a copy of document data when created.
PreviewMode	Set to “1” by the Web Server module if the preview parameter is requested in the URL. http://+:8080/webinputport/WebServer/?preview
PreviousJob	The PreviousJob JobInfo is set each time a job is cloned. It is primarily used by the Lasernet Monitor to figure out the sequence of jobs.
PrimaryJobData	PrimaryJobData constantly keeps the name of the JobInfo that contains the primary job data. Usually that is the JobData JobInfo. By changing the value of PrimaryJobData you change the JobInfo that Lasernet typically uses when it handles data. This can be done instead of copying the contents of JobData to another JobInfo. Changing PrimaryJobData is more efficient.
PrintAttachment	The PrintAttachment JobInfo array is used by the Printer Output and Printer Service modules. It may contain a list of documents in binary representation for printing externally. It is used together with PrintAttachmentFilename and PrintAttachmentMimeType JobInfos.
PrintAttachmentCopies	PrintAttachmentCopies setting is used by Printer Output and Printer Service modules and can be set to control the number of printed copies for attached documents in the format PDF and DOCX.
PrintAttachmentDocName	PrintAttachmentDocName is used by the Printer Output and Printer Service modules to acompany PrintAttachmentFilename and PrintAttachmentMimeType to give the print attachment a custom name in the Windows Spooler System.
PrintDocName	PrintDocName is used by the Printer Output and Printer Service modules to accompany PrintFilename and PrintMimeType to give the print job a custom name in the Windows Spooler System.
PrinterDriver	The PrinterDriver JobInfo is a configuration specific JobInfo often used by the EMF2RAW and Printer Output modules, if the JobInfo substitution string #PrinterDriver# is defined as the Printer Name parameter.
PrinterFailureMessage	PrinterFailureMessage contains the error message, if printing fails due to a Printer Failure Profile, attached to a Printer Output module.
PrinterName	The PrinterName JobInfo is a configuration specific JobInfo, often defined in modules, if the JobInfo substitution string #PrinterName# is used as the name of the destination to the Printer Output module.

PrinterServiceAzureStorageAccountName PrinterServiceAzureStorageContainerName PrinterServiceAzureStorageSASToken	The Printer Service module can be configured to read connection settings (for the Shared Access Signature (SAS) authentication option for Azure Storage) from JobInfos. This is an alternative to specifying the connection settings directly on the **Shared Access Signature Token (SAS)** tab of the **Add Print Server** window (in the Lasernet Config web app). If **Overridable** is selected on the **Shared Access Signature Token (SAS)** tab of the **Add Print Server** window (in the Lasernet Config web app), and any of the following three JobInfos exist, the value of the JobInfos that are present will override the corresponding print server settings' values in the Lasernet Config web app: • **PrinterServiceAzureStorageAccountName:** The name of the storage account that the Printer Service needs to access. This JobInfo overrides the print server's **Storage account** setting. • **PrinterServiceAzureStorageContainerName:** The name of the container that the Printer Service needs to access. This JobInfo overrides the print server's **Container name** setting. • **PrinterServiceAzureStorageSASToken:** The storage account SAS token. This JobInfo overrides the print server's **SAS Token** setting.
PrintFilename	Used by the Printer Output and Printer Service modules to extract and print jobs stored in a LnJob container. LnJob files are located in the runtime folder, as paused, scheduled or failed jobs. LnEMF and PDF documents are supported. The PrintFilename must be defined with the extension of the job stored in the LnJob container, like test.lnemf or test.pdf, to ensure that the document is printed in the expected format by the Printer Output module. This JobInfo is used combined with the PrintMimeType JobInfo.
PrintMimeType	The PrintMimeType JobInfo is used by the Printer Output and Printer Service modules. It contains the mime type of the file to be printed. Only used to accompany PrintFilename. The MIME types application/Lasernet and application/pdf are supported.
PrintMode	Set to "1" by the Web Server module if a print is requested.
PrintProcessor	Set by the Printer Input module. Contains the name of the print processor that handled the job.
PrintToUNC	Used by Printer Output module to direct jobs to a printer using a UNC path instead of a predetermined printer in Windows. By setting this JobInfo to \\Server\PrinterShare the Printer Output module will send the job to that share using the driver chosen in the setup.
Priority	Set by the Printer Input module. It contains the priority of the print job.
ProcessTime	Created by the Process Modifier. Contains the approximate number of seconds that the external application runs for. Since most applications only run for a short time this JobInfo is usually set to 0 (zero).
PublicID	The PublicID JobInfo is another unique ID for a Job. The PublicID is different from the JobID in that it follows all clones of the job, whereas JobID is unique for each clone. The PublicID is usually assigned in an input module.
RecognizedForm	The RecognizedForm JobInfo is created by the Form, Overlay and XML Transformer modules. The value is set to the name of the form that is recognized.
RecognizedSubJob	The RecognizedSubjob JobInfo is set by the Form Engine. The value is set to the number of the current input page within the entire job.
RecordCount	The RecordCount JobInfo is set by the Database Command when it returns records from a database. It is set to the number of records actually returned from the database. If it is 0 (zero) no records have been returned.

RequestHeaderNames RequestHeaderValues	A pair of JobInfo lists that contain the names and values of the HTTP headers in a request received by the Web Server input module. Each header's name is an item in RequestHeaderNames. That header's value is the corresponding item in RequestHeaderValues. For example, if the received HTTP request has a header `Content-Length: 11`, then `RequestHeaderNames[0]` = `Content-Length` and `RequestHeaderValues[0]` = `11`. Other headers received in the request are added as discrete items in the RequestHeaderNames and RequestHeaderValues JobInfo lists.
Restarted	The Restarted JobInfo is set when a failed job is restarted.
Scale	Scale is used by EMF2RAW and Printer Output modules. Specifies the factor by which the printed output is to be scaled. The apparent page size is scaled from the physical page size by a factor of Scale /100.
ScaleMode	Specifies the factor by which the printed output is to be scaled, for PDF documents stored in the JobInfo PrintAttachment. Legal values are: • Fit (Shrinks the size to fit the printer margins) • ActualSize (Default) • CustomScale (Set value for CustomScale (example: 90) as a procentage of the physical page. Modules: EMF2RAW, Printer Output, Printer Service
SchedulerQueue	The SchedulerQueue JobInfo is set by the scheduler on an engine or output module. It contains the name of the queue in which the job is placed. It is also set by the Scheduler Input module.
SFVFileName	Used to override the default name of the pre-uploaded checksum file. The ".sfv" file extension indicates a checksum file containing 32-bit CRC32 checksums in simple file verification format.
Sheet	Set by the Form Engine and Editor to the name of the sheet currently being processed (same as JobInfo for SheetName).
SheetName	Set by the Form Engine and Editor to the name of the sheet currently being processed (same as JobInfo for Sheet).
Source	The Source JobInfo is continuously updated with the names of the modules that the job has passed through.
StdOut	Created by the Process Modifier. Contains the data output to stdout from the external application.
TimeOut	The TimeOut JobInfo is created by the File Input module. The value is set to the time out value entered in the setup when using the JobInfo Scanner feature.
TransparentText	If your overlay does not have a white background this jobinfo must be set to make the background of the text box transparent. Valid values are: '1' and 'yes'. Only applies to the overlay engine.
UserDomainName	Create by Meta Input. Domain name for the user running Lasernet Meta.
UserName	Created by Meta Input and Printer Input modules. Contains the name of the sender in Lasernet Meta or the user who printed the job to a Lasernet Input Printer.

WebServiceCustomBody	Set custom body section in XML format if simple parameters in the SOAP body are not enough for the web service call. <?xml version="1.0" encoding="utf-8" ?> <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Header> <test xmlns="http://www.webservicex.net/"> <header>Header Section</header> </test> </soap:Header> <soap:Body> → Value of WebServiceCustomBody is inserted here ← <GetGeoIP xmlns="http://www.webservicex.net/"> <IPAddress /> <GetGeoIP /> </GetGeoIP> </soap:Body> </soap:Envelope>
WebServiceCustomHeader	Set custom header section in XML format if simple parameters in the SOAP header are not enough for the web service call. <?xml version="1.0" encoding="utf-8" ?> <soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"> <soap:Header> → Value of WebServiceCustomHeader is inserted here ← <test xmlns="http://www.webservicex.net/"> <header>Header Section</header> </test> </soap:Header> <soap:Body> <GetGeoIP xmlns="http://www.webservicex.net/"> <IPAddress /> <GetGeoIP /> </GetGeoIP> </soap:Body> </soap:Envelope>
WebServiceDetail	SOAP Fault detail. The detail element is intended for carrying application specific error information related to the Body element. The absence of the detail element in the Fault element indicates that the fault is not related to the processing of the Body element.
WebServiceFaultCode	SOAP Fault code. The fault code element is intended for use by software to provide an algorithmic mechanism for identifying the fault.
WebServiceFaultString	SOAP Fault string. The fault string element is intended to provide a human readable explanation of the fault and is not intended for algorithmic processing.
WebServiceResult	Default result JobInfo when used as modifier
WebserviceMethod	Name of Web Service method that corresponds to the HTTP request.
WinJobId	The WinJobId JobInfo is created by Printer Input module. The value is the job number assigned to the print job by the Windows Spooler.
WinPrinterName	The WinPrinterName JobInfo is created by the Printer Input module. The value is the name of the printer that was used by the input module.
WinPrintNotifyName	Specifies the notification contact of the print job. Windows does not allow the notify name to be set across network printers. Supported by Printer Output and Printer Service modules.
WinPrintUserName	Specifies the user name of the print job as shown in the job list for a local printer. This can be set to just about anything. Windows does not allow the user name to be set across network printers. Supported by Printer Output and Printer Service modules.
XMLValidationError	When running XML validation against a schema file, the XML Validation module will set this JobInfo to '1' if a validation has not been completed successfully.

XMLValidationErrorMessage	Description of the error that occurred in the XML Validation module when running XML validation against a schema file.

15 Regular Expressions.

15.1 Introduction

Lasernet supports regular expressions in several modules. It is mostly used in the Text Filter Modifier/Engine and Binary Filter Modifier, but it can also be very useful in JavaScript.

Regular expressions are built up from expressions, quantifiers and assertions. The simplest form of expression is simply a character e.g., **x** or **5**. An expression can also be a set of characters. For example **[ABCD]** will match an **A**, **B**, **C** or **D**. In shorthand it could be written as **[A-D]**. If we want to match any of the capital letters in the English alphabet we can write **[A-Z]**. A quantifier tells the regular expression engine how many occurrences of the expression we want e.g., **x{1,1}** means match an **x** which occurs at least once and at most once. Assertions and more complex expressions will be discussed later.

In general, regular expressions cannot be used to check for balanced brackets or tags. For example, if you want to match an opening html and its closing you can only use a regular expression if you know that these tags are not nested; the html fragment, bold bolder will not match as expected. If you know the maximum level of nesting it is possible to create a regular expression that will match correctly, but for an unknown level of nesting, regular expressions will fail.

We will start by writing a regular expression to match integers in the range 0 to 99. We require at least one digit, so we start with **[0-9]{1,1}** which means match a digit just once. This regular expression matches integers in the range 0 to 9. To match one or two digits we can increase the maximum number of occurrences. The regular expression then becomes **[0-9]{1,2}** meaning match a digit at least once and at most twice. However, this regular expression will not match correctly as its scope needs to be properly defined. To ensure that we match against the whole string we must use anchor assertions. By placing a **^** (caret) symbol first, the regular expression must match from the beginning of the string. By placing a **$** (dollar) symbol last, the regular expression must match until the end of the string. So our regular expression now looks like this ^**[0-9]{1,2}$**. Note that assertions such as **^** and **$**, do not match any actual characters.

If you have seen regular expressions elsewhere, they may have looked different from the ones above. This is because some sets of characters and some quantifiers are so common that they have special symbols to represent them. **[0-9]** can be replaced with the symbol **\d**. The quantifier to match exactly one occurrence, **{1,1}**, can be replaced with the symbol itself. This means that **x{1,1}** is the equivalent of just using **x**. So our 0 to 99 matcher could be written **^\d{1,2}$**. Another way of writing it could be **^\d\d{0,1}$**, i.e., from the start of the string match a digit followed by zero or one digits. In practice most people would write it **^\d\d?$**. The **?** is a shorthand for the quantifier **{0,1}**, i.e., a minimum of no occurrences and a maximum of one occurrence. This is used to make an expression optional. The regular expression **^\d\d?$** means “from the beginning of the string match one digit followed by zero or one digit and then the end of the string”.

Our second example matches the words 'mail', 'letter' or 'correspondence' without matching 'email', 'mailman', 'mailer', 'letterbox' etc. We'll start by just matching 'mail'. In full, the regular expression is **m{1,1}a{1,1}i{1,1}l{1,1}**, but since each expression itself is automatically quantified by **{1,1}** we can simply write this as **mail**; an 'm' followed by an 'a' followed by an 'i' followed by an 'l'. The symbol '|' (bar) is used for alternation, so our regular expression now becomes **mail|letter|correspondence** which means match 'mail' or 'letter' or 'correspondence'. Whilst this regular expression will find the words we want, it will also find words we don't want such as 'email'. We will start by putting our regular expression in parentheses **(mail|letter|correspondence)**. Parentheses have two effects: firstly they group expressions together and secondly they identify parts of the regular expression that we wish to capture. Our regular expression still matches any of the three words but now they are grouped together as a unit. This is useful for building up more complex regular expressions. It is also useful because it allows us to examine which of the words

actually matched. We need to use another assertion, this time **\b** “word boundary”: **\b(mail|letter|correspondence)\b**. This regular expression means “match a word boundary followed by the expression in parentheses followed by another word boundary”. The **\b** assertion matches a position in the regular expression not a character in the regular expression. A word boundary is any non-word character such as a space, a newline or the beginning or end of the string.

For our third example we want to replace ampersands with the HTML entity '&'. The regular expression to match is simple: **&**, i.e. match one ampersand. Unfortunately, this will ruin our text if some of the ampersands have already been turned into HTML entities. So what we really want to say is ‘replace an ampersand providing it is not followed by 'amp;”. For this we need the negative lookahead assertion. Our regular expression now becomes: **&(?!amp;)**. The negative lookahead assertion is introduced with '(?!' and finishes with ')'. It means that the text it contains, 'amp;' in our example, must not follow the expression that precedes it.

Regular expressions provide a rich language that can be used in a variety of ways. For example, suppose we want to count all the occurrences of 'Eric' and 'Eirik' in a string. Two valid regular expressions to match these are **\b(Eric|Eirik)\b and \bEi?ri[ck]\b**. We need the word boundary '\b' so we don't get 'Ericsson' etc. The second regular expression actually matches more than we want, 'Eric', 'Erik', 'Eiric' and 'Eirik'.

15.1.1 Characters and Abbreviations for Sets of Characters

Element	Meaning
c	Most characters represent themselves unless they have a regular expression function. Thus c matches the character c.
\c	A character that follows a backslash matches the character itself except where mentioned below. For example if you wished to match an actual caret at the beginning of a string you would write \^.
\a	This matches the ASCII bell character (BEL, 0x07).
\f	This matches the ASCII form feed character (FF, 0x0C).
\n	This matches the ASCII line feed character (LF, 0x0A, Unix newline).
\r	This matches the ASCII carriage return character (CR, 0x0D).
\t	This matches the ASCII horizontal tab character (HT, 0x09).
\v	This matches the ASCII vertical tab character (VT, 0x0B).
\xhhhh	This matches the Unicode character corresponding to the hexadecimal number hhhh (between 0x0000 and 0xFFFF). \0ooo (i.e., \zero ooo) matches the ASCII/Latin-1 character corresponding to the octal number ooo (between 0 and 0377).
. (dot)	This matches any character (including newline).
\d	This matches a digit (QChar::isDigit()).
\D	This matches a non-digit.
\s	This matches a whitespace (QChar::isSpace()).
\S	This matches a non-whitespace.
\w	This matches a word character (QChar::isLetterOrNumber() or '_').
\W	This matches a non-word character.
\n	The n-th back reference, e.g. \1, \2, etc.

15.1.2 Sets of Characters

Square brackets are used to match any character in the character set that is contained within them. All the character set abbreviations described above can be used within square brackets. Apart from the character set abbreviations and the following two exceptions, no characters have special meanings in square brackets.

Set	Description
^	The caret negates the character set if it occurs as the first character, i.e., immediately after the opening square bracket. For example, [abc] matches 'a', 'b' or 'c', but [^abc] matches anything except 'a', 'b' or 'c'.
-	The dash is used to indicate a range of characters, for example **[W-Z]** matches 'W', 'X', 'Y' or 'Z'.

The predefined character set abbreviations are more universal than using character ranges across platforms and languages. For example, [0-9] matches a digit in Western alphabets but \d matches a digit in any alphabet.

Note that in most regular expression literature, sets of characters are called “character classes”.

15.1.3 Quantifiers

By default, an expression is automatically quantified by **{1,1}**, i.e. it should occur exactly once. In the following list **E** stands for any expression. An expression is a character, an abbreviation for a set of characters, a set of characters in square brackets or any parenthesized expression.

Quantifier	Description
***E*?**	Matches zero or one occurrence of *E*. This quantifier means “the previous expression is optional” since it will match whether or not the expression occurs in the string. It is the same as ***E*{0,1}**. For example **dents?** will match 'dent' and 'dents'.
***E*+**	Matches one or more occurrences of *E*. This is the same as ***E*{1,MAXINT}**. For example, **0+** will match '0', '00', '000', etc.
E*	Matches zero or more occurrences of *E*. This is the same as ***E*{0,MAXINT}**. The * quantifier is often used by mistake. Since it matches zero or more occurrences, it will match when there are no occurrences at all. For example if we want to match strings that end in whitespace and use the regular expression **\s*$** we would get a match on every string. This is because we have said ‘find zero or more whitespace followed by the end of string’, so even strings that don't end in whitespace will match. The regular expression we want in this case is **\s+$** to match strings that have at least one whitespace at the end.
E{n}	Matches exactly n occurrences of the expression. This is the same as repeating the expression n times. For example, **x{5}** is the same as **xxxxx**. It is also the same as ***E*{n,n}** e.g., **x{5,5}**.
E{n,}	Matches at least n occurrences of the expression. This is the same as ***E*{n,**MAXINT**}**.
E{,m}	Matches at most m occurrences of the expression. This is the same as ***E*{0,m}**.
E{n,m}	Matches at least n occurrences of the expression and at most m occurrences of the expression.

(MAXINT is implementation dependent, but will not be smaller than 1024.)

If we wish to apply a quantifier to more than just the preceding character, we can use parentheses to group characters together in an expression. For example, **tag+** matches a 't' followed by an 'a' followed by at least one 'g', whereas **(tag)+** matches at least one occurrence of 'tag'.

Note that quantifiers are “greedy”. They will match as much text as they can. For example, **0+** will match as many zeros as it can from the first zero it finds e.g., '2.0005'. Quantifiers can be made non-greedy.

15.1.4 Assertions

Assertions make a statement about the text at the point where they occur in the regular expression, but they do not match any characters. In the following list ***E*** stands for any expression.

Assertion	Description
^	The caret signifies the beginning of the string. If you wish to match an actual ^ you must escape it by writing \^. For example, **^#include** will only match strings which begin with the characters '#include'. (When the caret is the first character of a character set it has a special meaning, see Sets of Characters.)
$	The dollar signifies the end of the string. For example, **\d\s*$** will match strings which end with a digit optionally followed by whitespace. If you wish to match an actual $ you must escape it by writing \$.
\b	A word boundary. For example, the regular expression **\bOK\b** means match immediately after a word boundary (e.g. start of string or whitespace) the letter 'O' then the letter 'K' immediately before another word boundary (e.g. end of string or whitespace). However, note that the assertion does not actually match any whitespace so if we write **(\bOK\b)** and we have a match, it will only contain 'OK' even if the string is “Its OK now”.
\B	A non-word boundary. This assertion is true wherever **\b** is false. For example if we searched for **\Bon\B** in “Left on” the match would fail (space and end of string are not non-word boundaries), but it would match in “tonne”.
(?=*E*)	Positive lookahead. This assertion is true if the expression matches at this point in the regular expression. For example, **const(?=\s+char)** matches 'const' whenever it is followed by 'char', as in 'static const char *'. (Compare with **const\s+char**, which matches 'static const char *'.)
(?!*E*)	Negative lookahead. This assertion is true if the expression does not match at this point in the regular expression. For example, **const(?!\s+char)** matches 'const' except when it is followed by 'char'.

16 Developing Web Services.

16.1 Web Service (Server)

A Web service is a way for two machines to communicate over a network in a client-server model. It has an interface (API) which is described in a machine-processable format called Web Services Description Language (WSDL). The machine hosting the web service is a server and the machine interacting with the server is a client.

16.1.1 Web Service Interface

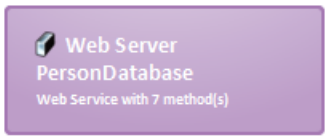
The Web service interface consists of one or more methods (subroutines/procedures/functions). A method has a range of zero or more input parameters and can return a result. A method defines the behaviour which Lasernet exhibits when called by a client.

The parameters and result of a method can use a range of built-in data types (string, integer, boolean etc). It is also possible to define custom data types.

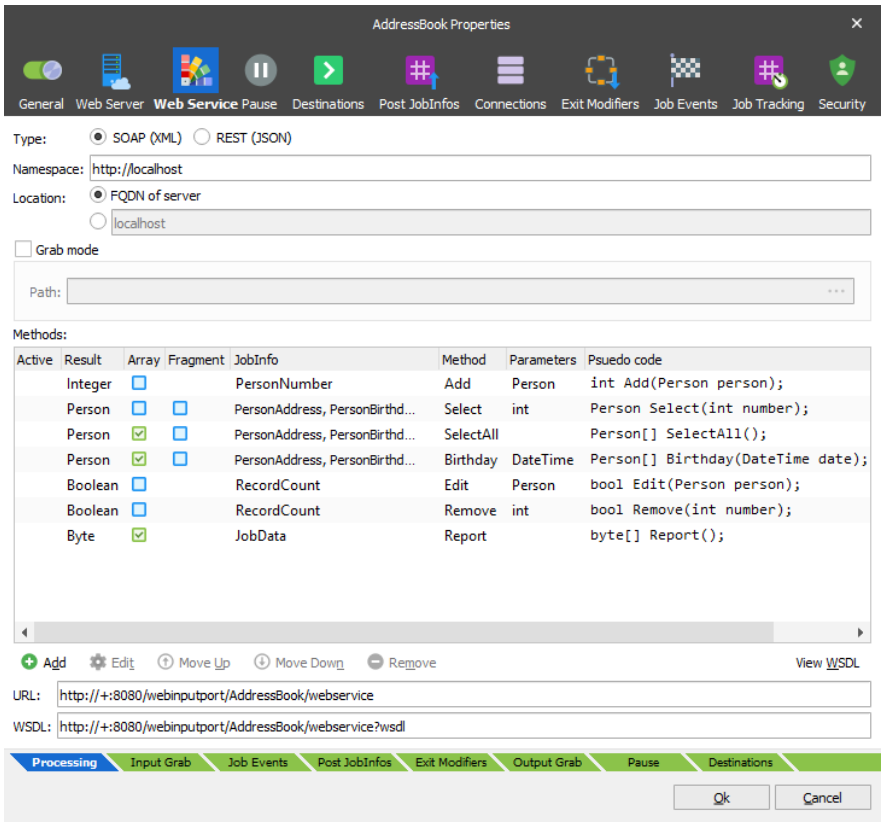
16.1.2 Lasernet as a Web Service server

In Lasernet it is possible to create your own SOAP Web services without having to implement a module using the SDK. One or more interfaces/API's can be defined by adding multiple Web Server input modules to a configuration. A Web Server object defines the API given by the name of the object.

Example of a PersonDatabase API:



APIs contain various methods to interact with the database:



16.1.3 Data Types

A Data Type can either be a record, with is a collection of variables, or an enumeration which consists of a set of named integral constants that are known as enumerators.

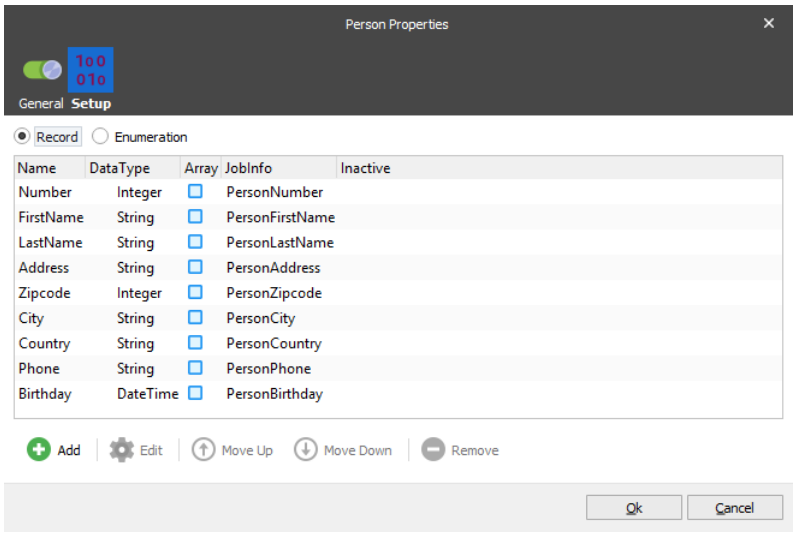
16.1.3.1 Record

For records, Lasernet supports a range of standard data types;

- **String**. A sequence of characters.
- **Byte**. An integer value between 0 and 255.
- **Integer**. An integer value between -2147483648 and 2147483647.
- **Boolean**. True/False value. ‘True’ and ‘1’ is considered true and everything else is false.
- **Double**. ±5.0 × 10−324 to ±1.7 × 10308. Precision of 15-16 digits.
- **DateTime**. From 12:00:00 midnight, January 1, 0001 Anno Domini (Common Era) through 11:59:59 P.M., December 31, 9999 A.D. (C.E.).

Lasernet also has a special data type called **JobInfo** which contains a key/value pair.

It is also possible to define custom data types, like ‘Person’ in our example:

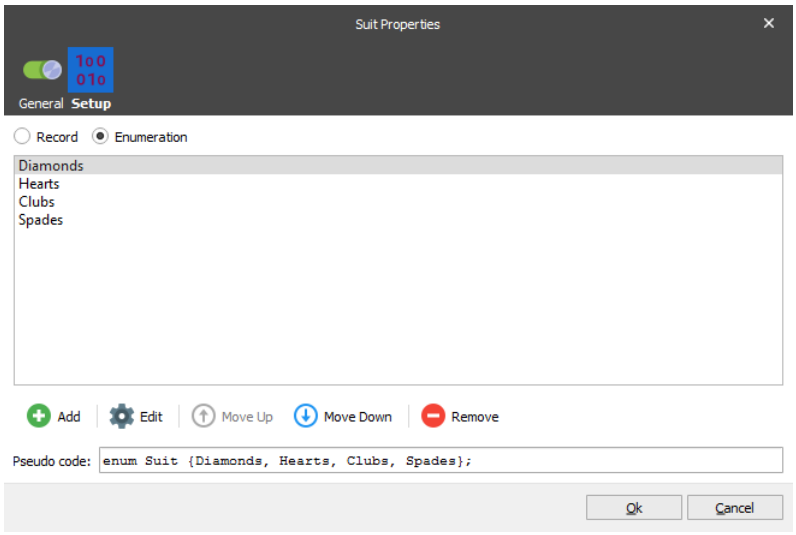


The ‘Person’ data type contains several variables; Number, FirstName, LastName, Address, Zipcode, City, Country, Phone and Birthday which have different data types.

To store binary data, like a file, an array of bytes can be used. The JobData of a Job in Lasernet is a byte array.

16.1.3.2 Enumeration

An enumeration is simply a list of strings. The enumeration can be used for both the parameters of a method and for the result.



16.1.4 Parameters

Each method in the API has its own parameter profile. Parameters can be a simple standard data type (string, integer etc.) or a custom user defined data type like ‘Contact’.

Reference Parameter

Lasernet supports reference parameters, which is basically a way of outputting data, if the method result is not enough. Only the simple standard data types can be used as reference parameters.

Array Parameters
Parameters can be marked as ‘array’ to allow the input of multiple values of the same data type.

JobInfo Data Type
Passing metadata to Lasernet can be done via the JobInfo data type. Passed JobInfos are not mapped to any specific JobInfo but use the key of the passed JobInfo. Be sure not to overwrite or append to existing JobInfos. Use a prefix where possible. Also take note that keys are not Lasernet system keys (for example JobID, PublicID etc).

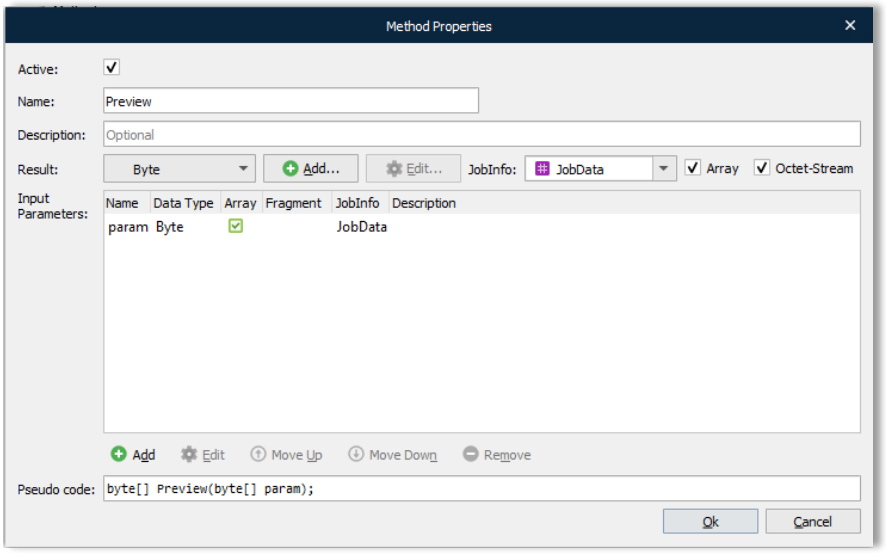
Complex Data Type
When returning a custom data type from Lasernet, this is considered a complex data type in some cases. The result itself can be an array, or the result can contain multiple arrays. These can be custom data types also.

Complex Data Types can be handled in two ways. Either save the data structure as an XML fragment, or map JobInfos into the structure if possible.

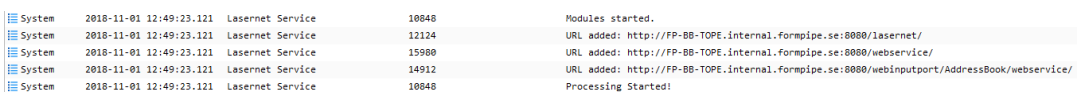
An XML fragment can be used as a template in the Form Editor, allowing the user to build up a complex result. This result must be placed in the JobInfo that is returned from the method when the Fragment flag is checked.

16.1.5 Result
Each method has its own result. This result can either be nothing (none or void), or a built in / custom data type.

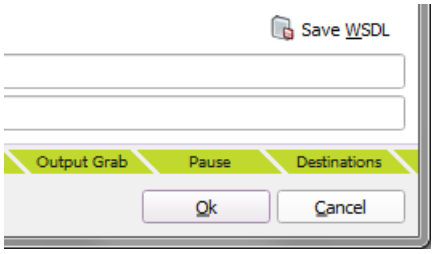
Lasernet is able to return a list of results in an array as seen in the ‘Birthday’ method:

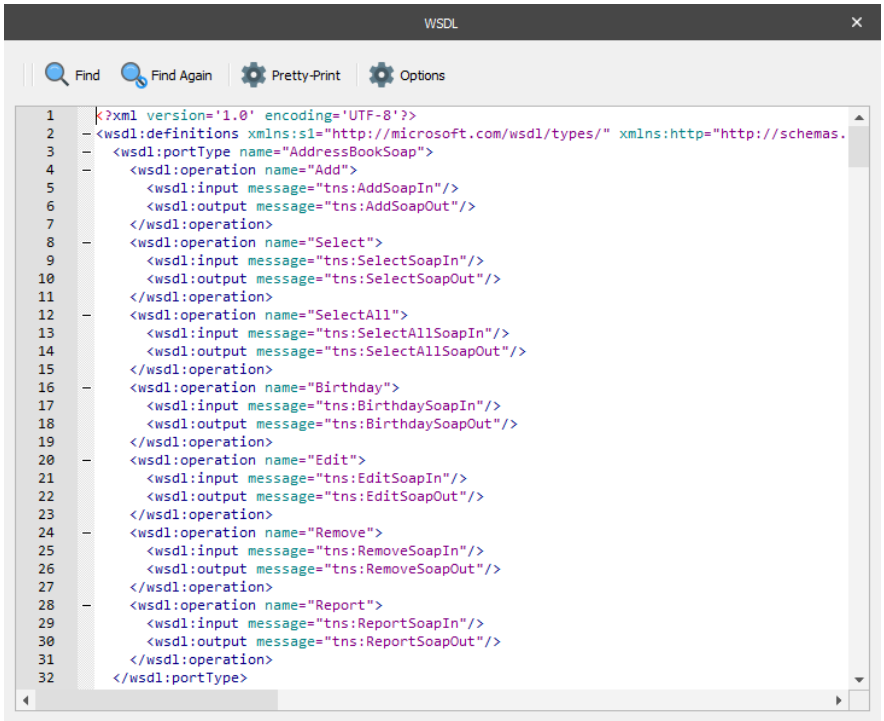


16.1.6 What happens in Lasernet when a method is called
After transferring the configuration containing the defined API, the log will show that the web server is ready to process requests:



The WSDL file of the Web service can be downloaded or referenced in the client software; either via the url with ‘wsdl’ query parameter, or by pressing the ‘Save WSDL’ button in GUI.





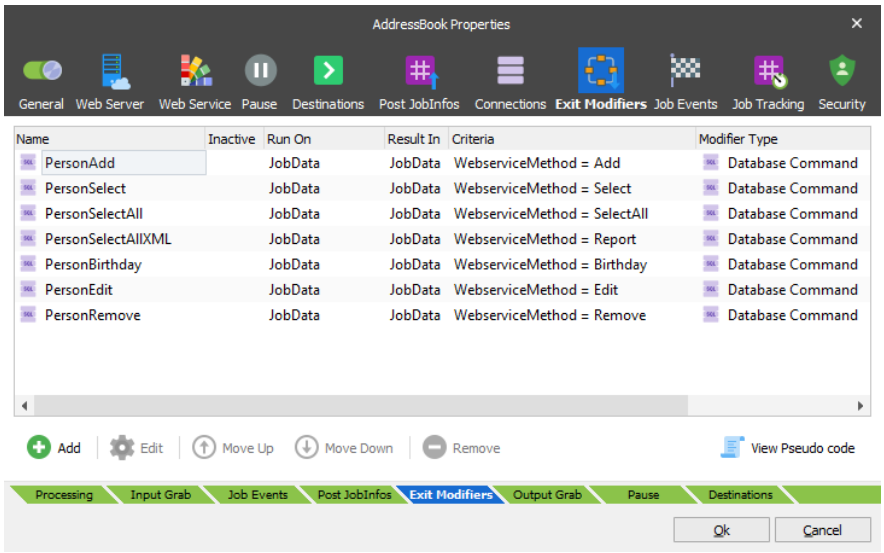
When Lasernet receives a request, a Lasernet Job is created and the method called is decoded from the SOAP request.

The method name is written to the ‘WebserviceMethod’ JobInfo of the Job.

Passed parameters are read into the mapped JobInfos of the Job.

The Job is then passed to the modifiers and destinations of the Web Server module.

Some methods might need to do advanced logic and processing. For this, the scripting functionality of Lasernet can be handy. Script functions can be called as modifiers, working as method handlers using the JobInfo criteria on the ‘WebserviceMethod’ JobInfo.



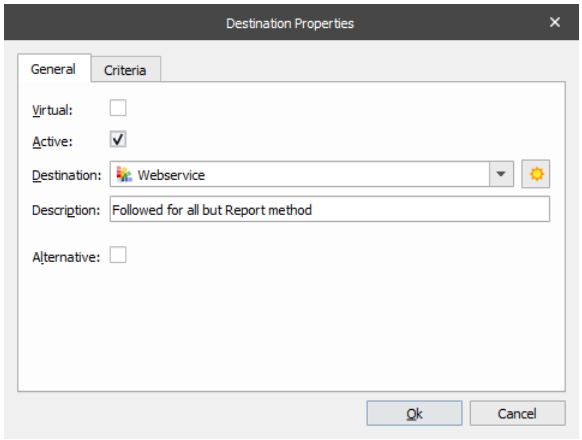
16.1.7 How and when does Lasernet respond to the client

Since web service method calls are blocking, it is important that Lasernet responds as soon as possible.

Lasernet is told to generate the response to the client, when a Job is passed to a special destination called ‘Webservice’. This is required and must be done as soon as possible to allow caller to block for as short a period as possible or avoid timing out.

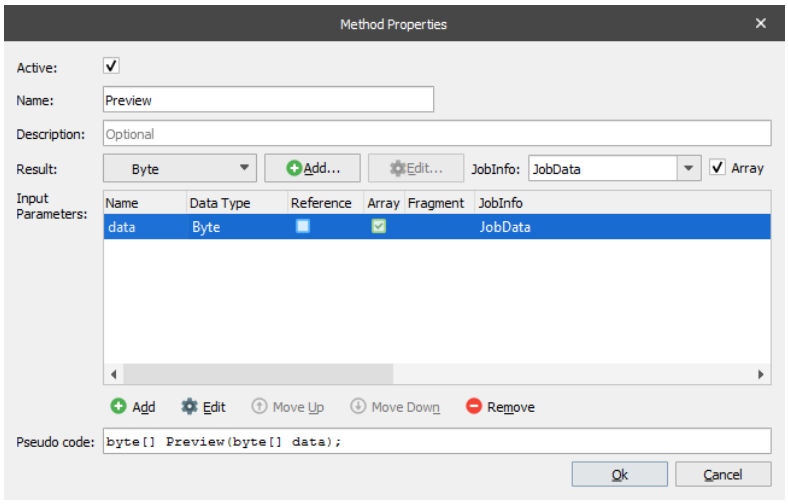
The virtual ‘Webservice’ destination can be called from either the Destinations list of the object, or from script via;

job.addDestination("Webservice");

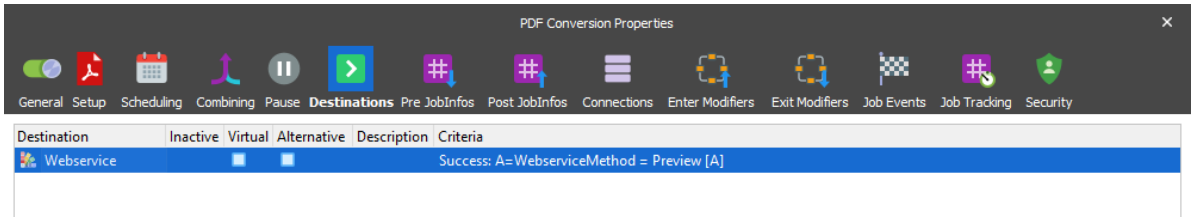


Any JobInfos needed for the response must be set before passing the Job to the ‘Webservice’ destination.

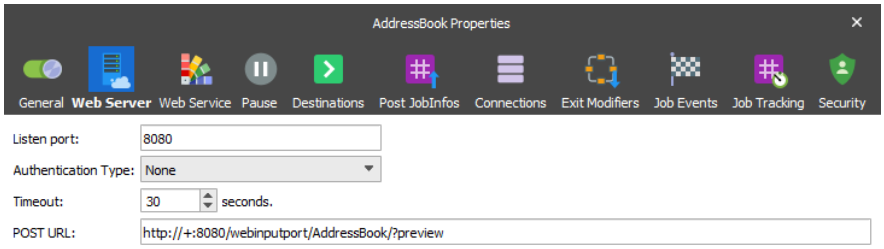
For example, a preview method generating and returning a PDF file;



This will return as soon as the PDF has been generated by the PDF engine.



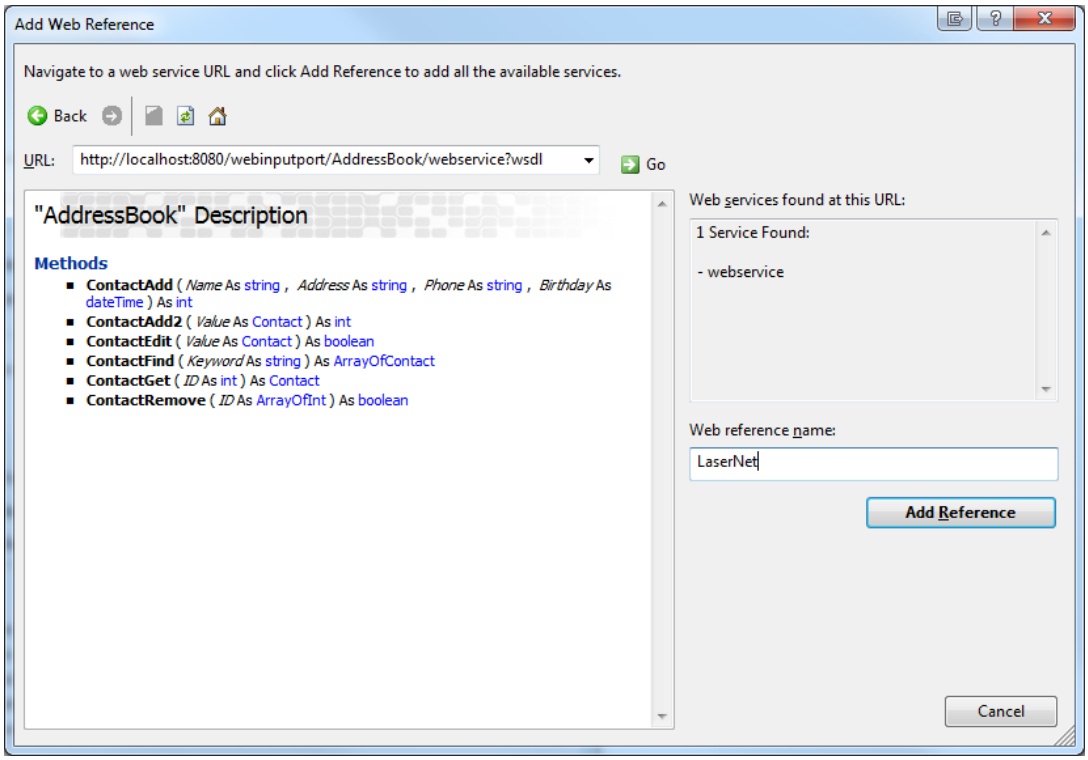
A timeout is defined both in the client and the server. Lasernet has a default and configurable 30 second timeout. If the Job has not been passed to the 'Webservice' destination within 30 seconds of receiving the request, a SOAP fault will be returned to the client.



16.1.8 Consuming the Web service

The Web service can be used from any client which is able to communicate via web services. Lasernet itself can be a client via the Web Service module.

Using .NET to consume the web service can be a simple, yet powerful way of integrating with Lasernet.



Notice how Visual Studio finds the exposed methods, their parameter profiles and generates code which is able to talk to Lasernet.

Calling a method in the Web service is now as simple as a few lines of code;

```
LaserNet.AddressBook ws = new LaserNet.AddressBook();

ws.Url = "http://localhost:8080/webinputport/AddressBook/webservice";


int contactID = ws.ContactAdd(
   /*Name*/"Jacob Pedersen",
   /*Address*/"South of Heaven",
   /*Phone*/"1234",
   /*Birthday*/DateTime.Parse("1974-08-18T11:11")
   );
```

16.1.9 Combining multiple Jobs into a single web service response

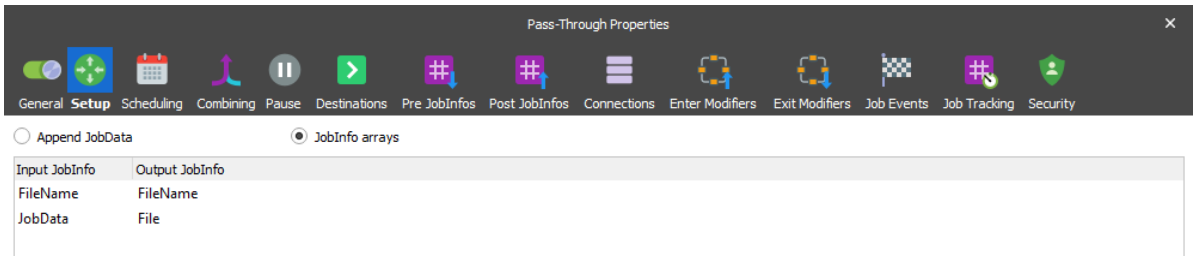
If a method needs to return multiple processed Jobs in a single request, it is necessary to combine these Jobs into a single Job before sending that to the 'Webservice' destination.

The ‘Webservice’ destination uses the PublicID of the incoming request to know where to send the response. This means that multiple calls to the ‘Webservice’ destination, with the same PublicID, will result in an ‘Unknown destination’ error.

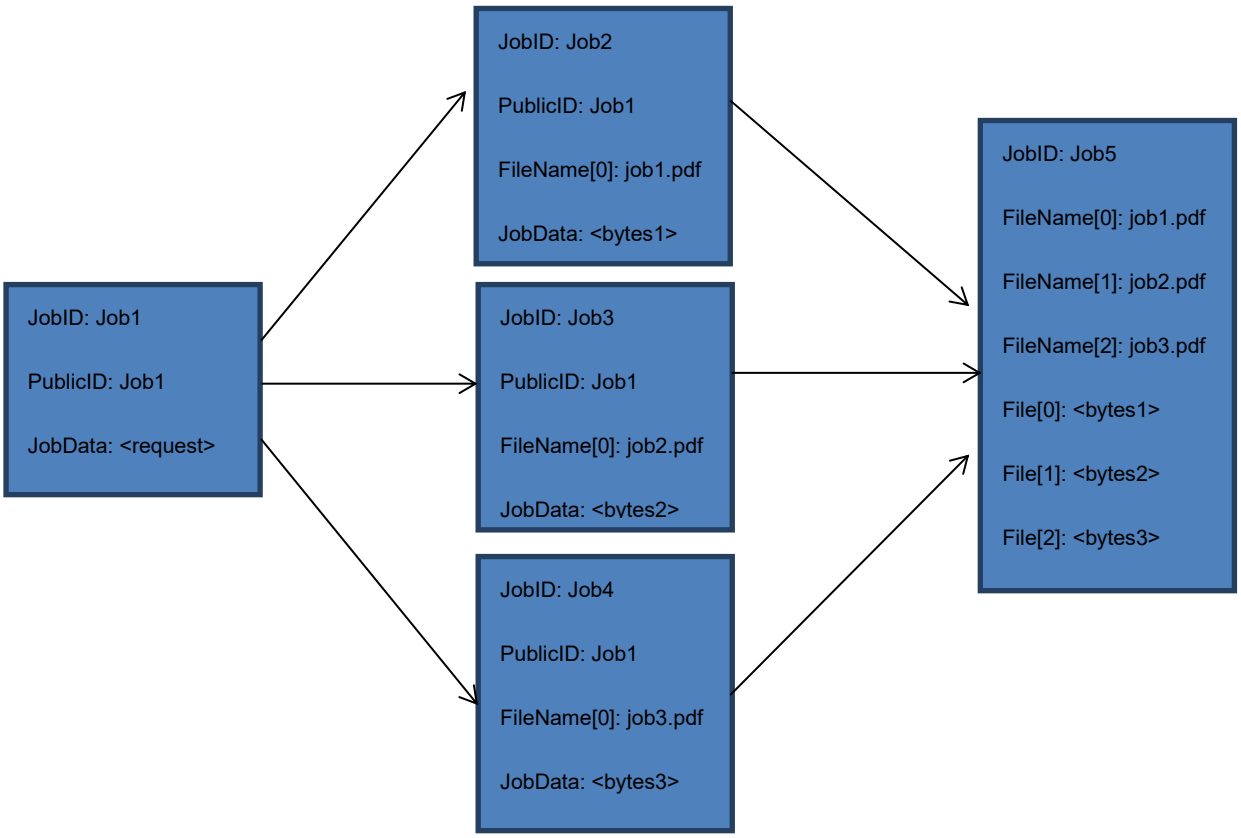
The way a method returns arrays is by using the JobInfo arrays. Therefore, we need to convert the combined Jobs into one Job, where JobInfos for each combined Job, are written to arrays on the combined outcoming Job.

The Pass-Through engine has this functionality. Enabling combining on a Pass-through engine will make it automatically combine on ‘PublicID’. If possible, set a combiner stop criterion, so Lasernet does not have to wait for a timeout. This will help Lasernet respond as soon as possible to the client.

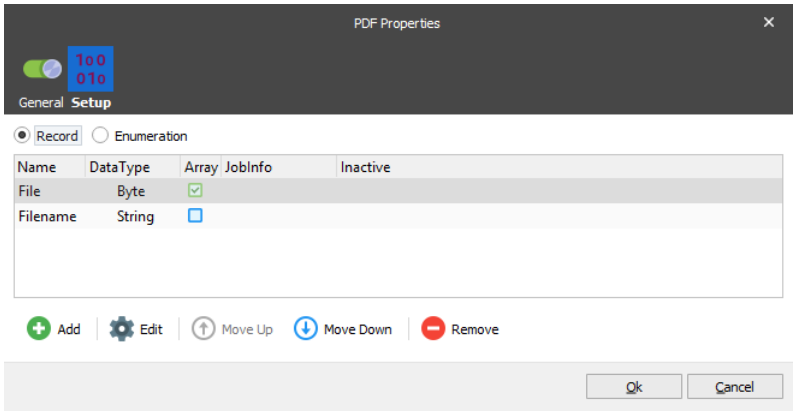
The outcoming combined Job from the Pass-Through engine, will be a clone of the first incoming Job. The JobInfo arrays radio button makes it possible to specify a list of JobInfos which will be copied from each incoming Job into arrays on the outcoming Job:



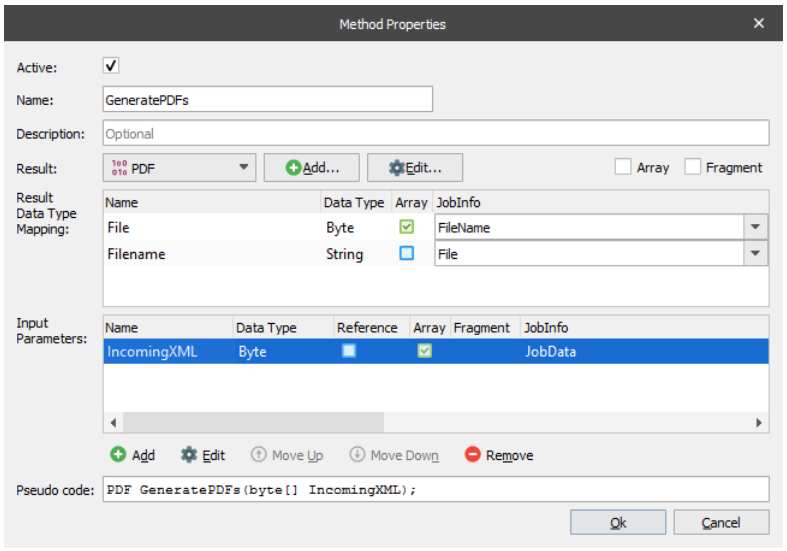
As an example, take a request which results in 3 incoming Jobs which will be combined into 1 outcoming Job to be passed to ‘Webservice’ destination:



The Data Type profile is defined as follows:



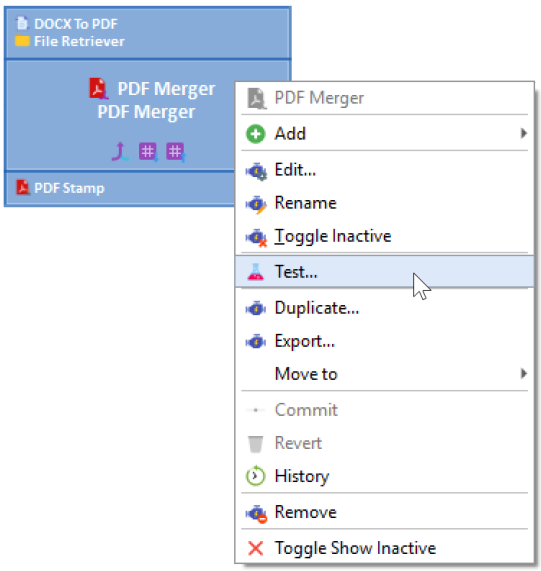
Settings for the Method to output an array of PDF's that maps the 2 JobInfo arrays from the Pass-Through engine:



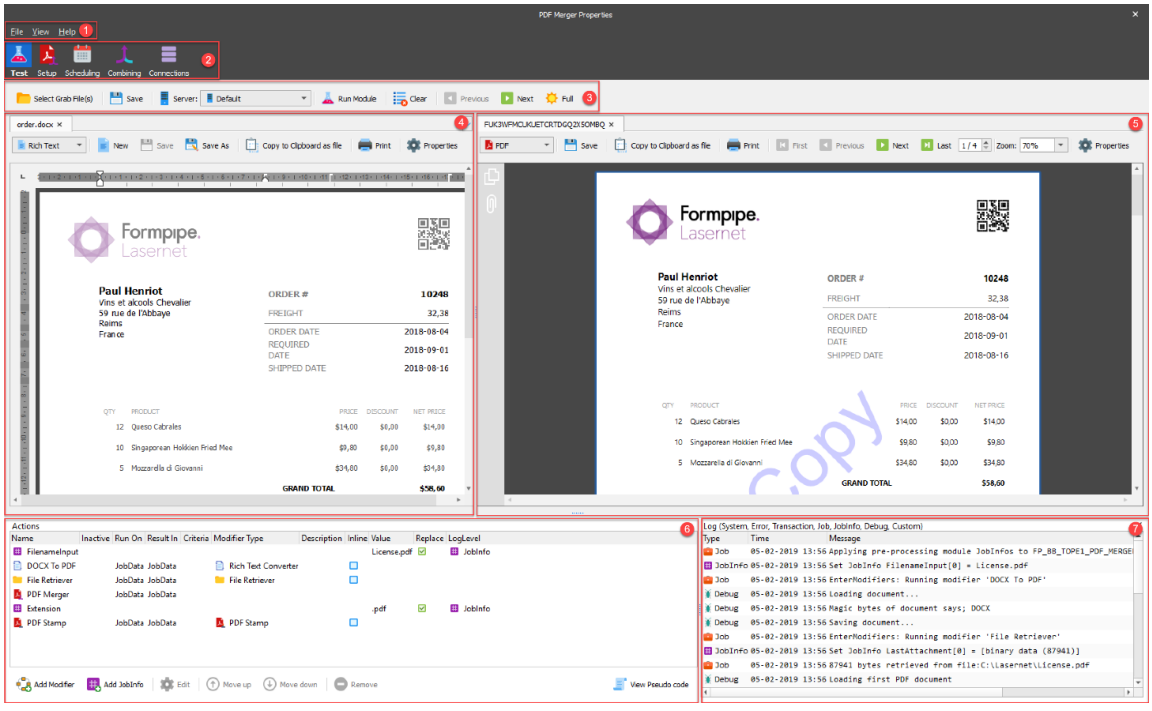
17 Lasernet Module Tester.

Run and verify module settings.

Right click on any module in the configuration and select **Test...** to run the Lasernet Module Tester.



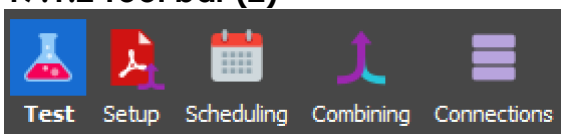
The Lasernet Module Tester is a built-in application, which can process incoming jobs and run the module similarly to the Lasernet Job Engine. Each action added to a module can be inspected one by one, and you can preview the contents of the output and all the log messages as you step through all the actions.



17.1.1 Menu bar (1)

From the **File** menu you can **Save** all the settings or **Close** the Lasernet Module Tester. From the **View** menu you can activate additional window views for JobInfos and Logger windows and customize the log events in the Logger window.

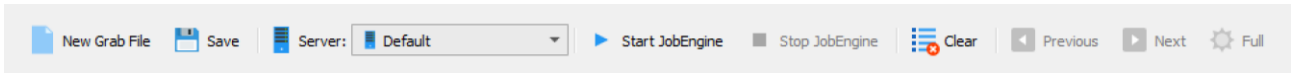
17.1.2 Tool bar (2)



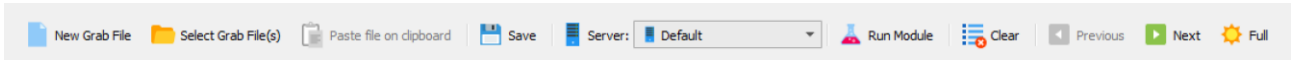
Select the **Test** button in the tool bar to show the contents of input- and output jobs, test buttons and action list. The **Setup**, **Scheduling**, **Combining** and **Connections** are standard buttons present in most modules and are used to configure job processing. The list of functions in the tool bar varies from module to module, depending on features and functions.

17.1.3 Action bar (3)

The Lasernet Module Tester has two different action bars depending on the type of module. An action bar for input modules, where the incoming test job(s) are retrieved, removed and processed directly from the input, which the input module is connected to. Job Processing is started when you click the **Start JobEngine** button.



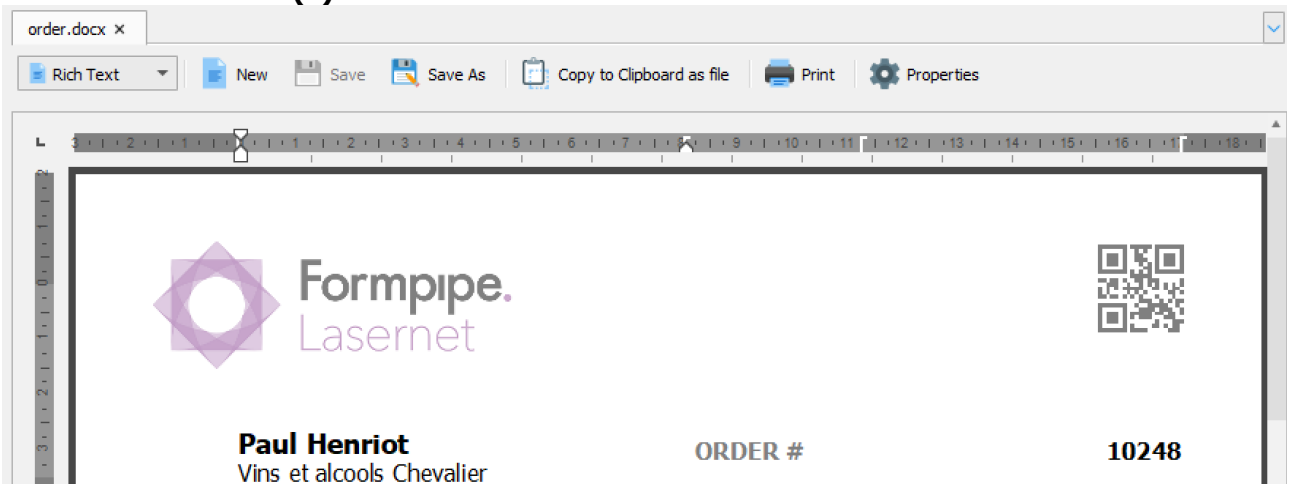
Another action bar is present for engines and output modules. Incoming test job(s) are selected from the **Select Grab File(s)** dialog and the **Run Module** button is used to process all settings and actions added to the module.



Click the **Clear** button to clear the logger window (if it is active). Use **Previous** and **Next** to step through the modifier and JobInfo actions one by one. Click the **Full** button to see changes from all actions.

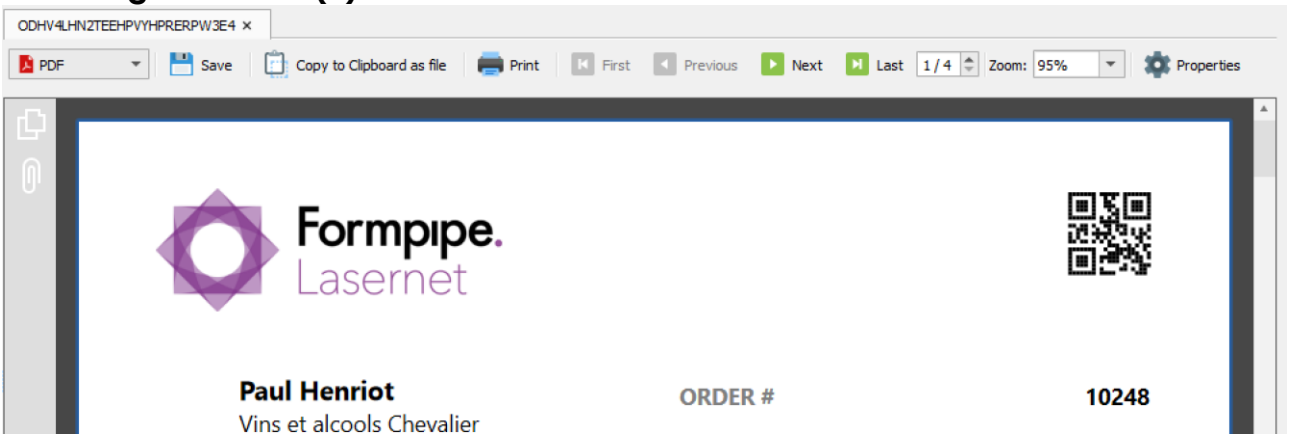
For output modules, the result of the test will be saved, posted or emailed depending on the module type, directly to the target defined in the settings of the module, similar to the Lasernet Service.

17.1.4 Left window (4)



The left preview window shows the contents of the grab data used for the test. A wide range of known preview formats are supported. None of the supported formats can be previewed in HEX mode or as text. Only Engines and Output modules can open grab files.

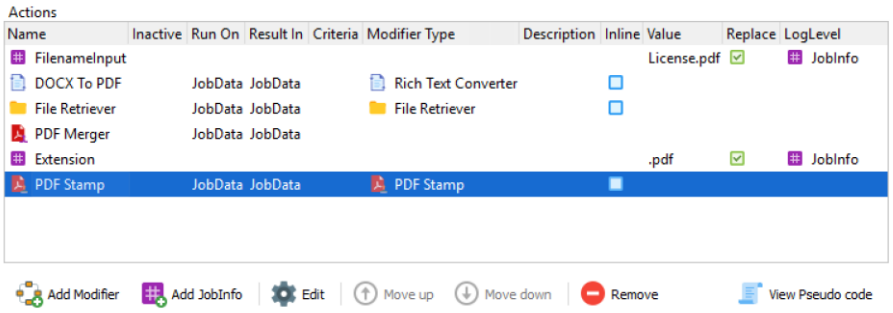
17.1.5 Right window (5)



The right preview window shows the results of the module test. The contents will change as you step through each action added to module.

For more detailed information about JobInfo values, assigned to the job, you can activate View → JobInfos.

17.1.6 Actions (6)



In the **Actions** list you can **Add** and **Edit** the modifiers and JobInfos assigned to the module or change the processing order by selecting an action and clicking the **Move up** and **Move down** buttons. Or you can click on any action in the list view, to inspect the result of all actions before the selected action.

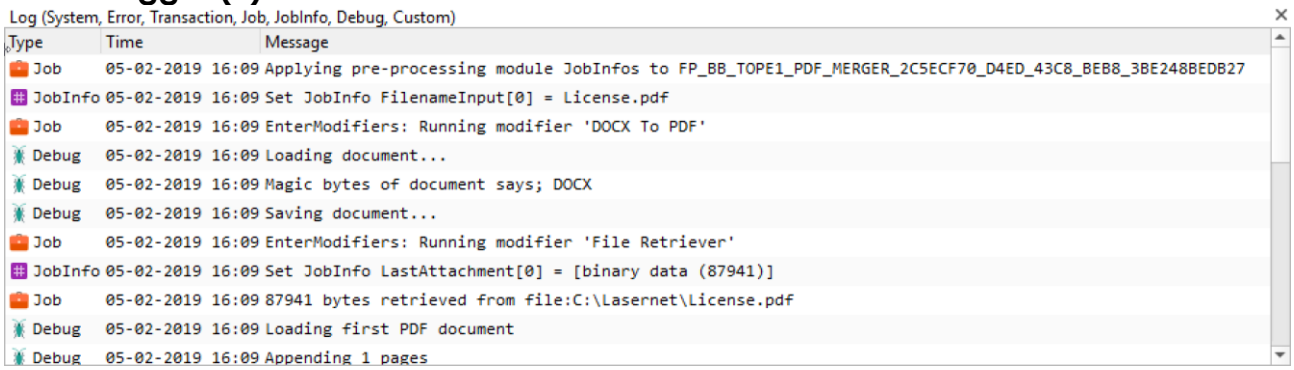
Click the **Remove** button to remove one or more actions from the action list.

Click **View Pseude code** for a quick review of all actions including criteria in text mode. Pseudo code is for reading only and cannot be edited.

When you have successfully tested the module, click **File → Save** to store all settings and actions added to the module or **File → Close** to ignore the current settings.

Changes to actions will require a rerun of the test to see the results.

17.1.7 Logger (7)



If the **Logger** window is active, you can view the full list of logs created for the tested module. Go to **View → Log Events** to specify the type of log events you want to view in test mode.

17.1.8 User rights

In **Test…** mode you must start the Lasernet Developer, with similar user rights as the Lasernet Service, to access your database connections, file shares etc., which you have defined in the module or in actions added to the module.

To run the Web Server Input module, you must start the Lasernet Developer with **Admin** rights or you will receive a “HttpAddUrl failed with error code: 5”, when you click **Start JobEngine**.

18 Windows Print Spooler and Printer Drivers.

18.1 Introduction to the Print Spooler

The print spooler is the Windows component that enables print jobs to be routed to local and network printers. The spooler runs as the Windows service 'Print Spooler' and performs the following tasks:

- Administers print job queues;
- Converts print files to printer compatible formats;
- Generates prints in a format understood by the printer;
- Sends the prints to the printer module;
- Reads status messages e.g. 'out of paper', from the printer module.

These tasks are handled by Print Providers, Print Processors, Printer Drivers, Port Monitors and Language Monitors, respectively.

18.2 Print Formats

A print file format describes the layout and contents of the pages to be printed. The file format is also known as the data type or the data format of the print job. You can see the print job format by pausing the printer, then selecting properties on the job and looking at the data type field.

The most important print formats are:

18.2.1 Enhanced Windows Metafile (EMF)

The native Windows spooler print format is the Enhanced Windows Metafile format. This is the default print format used by Windows applications and is also the format that printer drivers use as their input. All other formats have to be converted to EMF before they can be sent to a printer driver – a print processor usually does this.

18.2.2 Plain text (TEXT)

Plain text formats are simply text files with no formatting.

18.2.3 Printer Control Language (PCL)

PCL is the native format used by most Hewlett-Packard (HP) printers. Many other printer vendors also support this format and it is commonly used by non-Windows applications.

18.2.4 PostScript (PS or PSCRIPT)

PostScript is a page description language used primarily in the electronic and desktop publishing areas and on UNIX systems. Neither the Windows print spooler nor Lasernet have direct support for this format.

18.2.5 Portable Document Format (PDF)

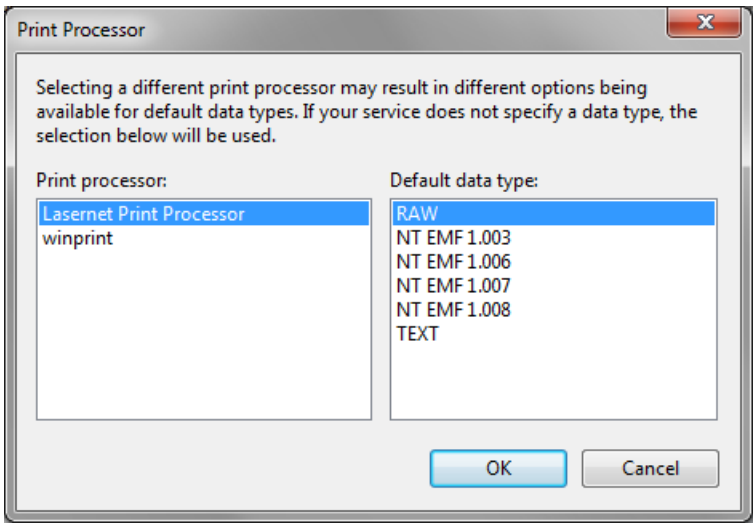
Portable Document Format is a file format developed by Adobe Systems for representing documents in a manner that is independent of the original application software, hardware and operating system used for creation of those documents. PDF is an extension of the PostScript format.

18.2.6 RAW

The RAW format is not actually a specific format, but means that the print is already in a format understood natively by the printer and should be sent to it as is.

18.3 Print Processors

The print processor is the part of the print spooler that is responsible for converting any given input into a format that is understood by the printer. Every printer in Windows has a default print processor and print format associated with it. To view these settings, select properties on the printer and click the **Print Processor** button on the **Advanced** tab.



Each print processor shows a list of formats that it recognizes and is able to convert into a printer compatible format. WinPrint, the default print processor, supports RAW, NT EMF and TEXT. If the printing application does not specify what format a print is provided in, the print spooler will default to the format and print processor selected on the printer. Unless changed on the printer, the default format is RAW. This means that if we copy a file to the printer i.e., copy /b c:\printfile \\mycomputer\myprintershare, the system will assume it is in RAW format.

The WinPrint print processor uses a printer driver for all its converting, except for RAW format. If data is received in RAW, the print processor passes it through unchanged. Otherwise, the print processor has to convert the remaining formats into EMF before they can be passed along to the printer driver. For formats like TEXT, WinPrint selects a Courier New font and assumes 80-character columns and 72 rows. Although many printers understand raw text files, the print processor will still convert it into EMF and pass it through the printer driver, to produce a print file in the printer's native format, such as PCL for HP printers.

The Lasernet Print Processor is different. It will never perform any format conversion, always passing the input print format as it is to the printer module. If an application or printing service, such as printing from a Unix system, informs the Windows print spooler that the print is in TEXT, the usual WinPrint print processor will deliver the printers native format to Lasernet. This can sometimes be fixed by changing the data type on the UNIX system. However, changing the print processor to the Lasernet Print Processor is an alternative way to solve the problem, as the Form Engine needs to work with text.

18.4 Printer Drivers

A printer driver is the part of the print spooler that converts EMF data into the native format of the printer. The choice of printer driver affects the output received by Lasernet whenever a Windows application prints, or when a print processor can convert the print into EMF and pass it to a printer driver.

Lasernet is bundled with the following printer drivers:

18.4.1 Lasernet Text Only

This driver generates flat formatted text files for Lasernet. It is the most commonly used driver for input printers and normally produces the best print for the Form Engine. Using the device settings on the printer, you can control how many rows and columns the driver maps the print into.

18.4.2 Lasernet EMF

The Lasernet EMF driver generates prints in the native Lasernet print format, LnEMF. The print produced includes everything printed by the application, including images and vector graphics. This is the ideal driver for generating overlays and when using the overlay engine. It is often used in combination with the Lasernet Print Processor to capture print as pure EMF and use LnEMF as a fallback.